# TURKISH TEXT GENERATION WITH SYSTEMIC-FUNCTIONAL GRAMMAR

A THESIS
SUBMITTED TO THE DEPARTMENT OF COMPUTER
ENGINEERING AND INFORMATION SCIENCE
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Turgay Korkmaz
June, 1996

# TURKISH TEXT GENERATION WITH SYSTEMIC-FUNCTIONAL GRAMMAR

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER

ENGINEERING AND INFORMATION SCIENCE

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Turgay Korkmaz

June, 1996

P
98
.K67
1996

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Asst. Prof. Dr. İlyas Çiçekli(Principal Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Dr. Halil Altay Güvenir

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Asst. Prof. Dr. Özgür Ulusoy

Approved for the Institute of Engineering and Science:

[illegible]

Prof. Dr. Mehmet Baray,
Director of Institute of Engineering and Science

## ABSTRACT

# TURKISH TEXT GENERATION WITH SYSTEMIC-FUNCTIONAL GRAMMAR

Turgay Korkmaz
M.S. in Computer Engineering and Information Science
Advisor: Asst. Prof. Dr. İlyas Çiçekli
June, 1996

Natural Language Generation (NLG) is roughly decomposed into two stages: *text planning*, and *text generation*. In the text planning stage, the semantic description of the text is produced from the conceptual inputs. Then, the text generation system transforms this semantic description into an actual text. This thesis focuses on the design and implementation of a Turkish text generation system rather than text planning. To develop a text generator, we need a linguistic theory that describes the resources of the desired natural language, and also a software tool that represents and performs these linguistic resources in a computational environment. In this thesis, in order to carry out the mentioned requirements, we have used a *functional linguistic theory* called Systemic-Functional Grammar (SFG), and the FUF text generation system as a software tool. The ultimate text generation system takes the semantic description of the text sentence by sentence, and then produces a morphological description for each lexical constituent of the sentence. The morphological descriptions are worded by a Turkish morphological generator. Because of our concentration on the text generation, we have not considered the details of the text planning. Hence, we assume that the semantic description of the text is produced and lexicalized by an application (currently given by hand).

*Keywords*: Natural Language Processing, Natural Language Generation, Computational Linguistic, Systemic-Functional Grammar, Functional Unification Grammar.

# ÖZET

## SİSTEMİK-FONKSİYONEL GRAMER YAKLAŞIMI İLE TÜRKÇE METİN ÜRETİMİ

Turgay Korkmaz
Bilgisayar ve Enformatik Mühendisliği, Yüksek Lisans
Danışman: Yrd. Doç. Dr. İlyas Çiçekli
Haziran, 1996

Doğal Dil Üretimi (DDÜ) kabaca iki kısıma ayrılır: *metin planlama* ve *metin üretme*. Metin planlama kısımında, kavramsal girdilerden metinin anlamsal tanımı üretilir. Sonra, metin üretme sistemi bu anlamsal tanımları gerçek bir metine dönüştürür. Bu tez metin planlamadan ziyade Türkçe metin üretecek bir sisteminin tasarım ve gerçekleştirimi üzerinde durmaktadır. Bir metin üretici geliştirmek için, doğal dilin kaynaklarını tanımlayacak bir dilbilim teorisine, ve bu kaynaları bilgisayar ortamında gösterecek ve işleyecek bir yazılım aracına ihtiyacımız vardır. Bu tezde, *Sistemik-Fonksiyonel Gramer* (SFG) olarak bilinen fonksiyonel dilbilim teorisini, ve yazılım aracı olarak da FUF metin üretme sistemini kullandık. Gerçekleştirilen metin üretim sistemi metinin anlamsal tanımını cümle cümle alıyor, ve cümledeki her bir sözcüksel öğenin şekil bilgisini üretiyor ki bunlar Türkçe sözcüklerin şekil bilgilerinden kelimeler üreten bir program tarafından kelimeleştirilmektedir. Metin üretimi üzerinde yoğunlaşmamızdan ötürü, metin planlama kısmını ayrıntılı olarak incelemedik. Bu yüzden, metinin anlamsal tanımının bir uygulama tarafından üretildiğini ve sözcüklendirildiğini kabul ediyoruz (şu an elle veriliyor).

*Anahtar sözcükler*: Doğal Dil İşleme, Doğal Dil Üretimi, Bilgisayarlı Dilbilimi, Sistemik-Fonksiyonel Gramer, Fonksiyonel Birleştirme Grameri.

# ACKNOWLEDGMENTS

I am very grateful to my supervisor, Assistant Professor İlyas Çiçekli for his invaluable guidance and motivating support during this study. His instruction will be the closest and most important reference in my future research.

I would also like to thank to Assoc. Prof. Dr. Halil Altay Güvenir and Asst. Prof. Dr. Özgür Ulusoy for reading and commenting on the thesis.

Finally, I would like to thank my family and everybody who has in some way contributed to this study by giving me moral support.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| 1SG, 2SG, 3SG | first, second, third person singular |
| 1PL, 2PL, 3PL | first, second, third person plural |
| 1SP, 2SP, 3SP | first, second, third person singular possessive |
| 1PP, 2PP, 3PP | first, second, third person plural possessive |
| ABL | ablative (*+dEyn*) |
| ACC | accusative (*+yI*) |
| ADJ | adjective |
| ADV | adverb |
| AdvG | adverb group |
| AOR | aorist (positive: *+Er* and *+Ir*; negative: *+z*) |
| APP | approximation suffix (*+yEyaz*) |
| CAUS | causative (*+dIr, +t*) |
| COMP | comparative |
| COND | conditional (*+sE*) |
| CONV=... | conversion to ... |
| COP | copula (*+dIr*) |
| DAT | dative (*+yE*) |
| DUR | durative suffix (*+yEdur, +yEkoy, +yEkal*) |
| FUT | future (*+yEcEk*) |
| GEN | genitive (*+nIn*) |
| HASTE | haste verb suffix (*+yIver*) |
| IMP | imperative |
| INF | infinitive (*+mEk*) |
| INS | instrumental (*+lE*) |
| LOC | locative (*+dE*) |
| MUN | munitive (*+lI*) |
| NARR | narrative past (*+mIş*) |
| NECES | necessitative (*+mElI*) |
| NEG | verbal negative (*+mE*) |

| | |
|---|---|
| NP | noun phrase (group) |
| OPT | optative (*+yE*) |
| PART | participle conversion |
| PASS | passive (*+In, +Il*) |
| PAST | past (*+dI*) |
| POT | positive potential (*+yEbIl*) |
| POTneg | negative potential (*+EmEm*) |
| PP | post-positional group |
| PRIV | privative suffix (*+sIz*) |
| PROG | progressive (*+Iyor*) |
| Ques | yes/no question (*mI*) |
| REL | relativization (*+ki*) |
| VG | verbal group |

# Chapter 1

# Introduction

Natural Language (NL) is a communication system that enables people to express their feelings, thoughts or demands as a sequence of words in a particular social and cultural environment, or vice versa. In other words, NL encodes a mental picture of reality into a sequence of words called a grammatical unit such as clause, noun group etc., or decodes the sequence of words into a mental picture (as shown in Figure 1.1). The encoding and decoding activities are

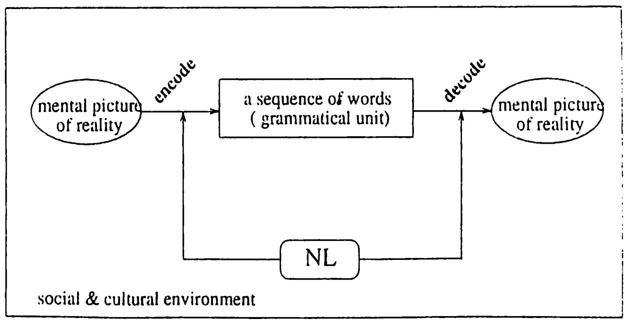
mental picture of reality
encode
a sequence of words ( grammatical unit)
decode
mental picture of reality
NL
social & cultural environment

Figure 1.1: Natural Language (NL) is a system

called *Generation* and *Understanding*, respectively. Natural Language Processing (NLP) is a research area that aims to simulate those human activities on computer because of several reasons that may be generalized as follows [30, 32]:

- To provide human–computer communication with a particular NL.
- To translate information from one NL to another via computer.

This thesis particularly deals with Natural Language Generation (NLG). From the computational perspective, NLG can be described as a process of constructing a text from information which is stored at a higher order of abstraction rather than wordings [4, 30, 32]. Certainly the bottom end of NLG is the worded text, but at the top end, the boundaries of abstraction are not easy to define exactly. To address the different kinds of problems, the NLG process is roughly decomposed into two stages: *text planning*, and *text generation* (realization). The text planner produces the semantic description of the generated text from the conceptual inputs. And then the text generator takes the semantic description as input, and transforms it into the worded text according to the linguistic resources of the desired natural language.

So far, several NLG systems have been constructed as parts of PhD theses such as Goldman's BABEL [12], Davey's PROTEUS [5], McDonald's MUMBLE [31], Mckeown's TEXT [33], Appelt's KAMP [1], Patten's SLANG [37], Hovy's PAULINE [16], Elhadad's SURGE [7].[1] Each of them tries to address the common problems in NLG from different perspectives. Therefore, a number of approaches to NLG have been introduced in those works. A NLG system and its approach can be characterized by:

- the organization of the text planning and generation stages,
- the linguistic theory that the system is based on,
- the computational formalisms that the system uses,
- the social context and the field of text generation.

According to the relation between the text planner and the generator (realizer), NLG systems are divided into three classes: *pipelined, interleaved,* and *integrated* (see [18]). In a pipelined system, the text planner produces the required information as input for the generator, and then the generator produces the worded text without any communication from the generator back to the planner. In contrast to the pipelined system, an interleaved system provides communication from the generator back to the planner. In an integrated system, planning and generation stages are considered in a single formalism.

A linguistic theory allows us to describe the resources of NL, and to construct a model that explains *how* NL transforms the semantic description into

[1]More information about the current NLG systems and their approaches can be found in [4, 7, 18, 30, 32, 37].

a grammatical structure and its lexical items. Most of the current theories are only interested in *syntax, morphology*, or *phonology*, but not *semantics* or *context* [30]. However, there is a strong relation between semantic and syntactic descriptions of NL. To represent those relations, we need a linguistic theory that analyzes the NL from both semantic and syntactic perspectives. In this context, the functional theories such as tagmemic theory, systemic theory and stratificational theory will be more appropriate than the structure based theories [30].

The computational formalisms provide different methods and notations for representing and performing the linguistic resources on computer without depending on the linguistic theory. This can be achieved by the distinction between the linguistic theory and the implementation formalism. Recently the unification and feature structures have been used as computational formalisms in the generation. One of them is *Functional Unification Formalism* (FUF) [7] derived from *Functional Unification Grammar* (FUG) [22], and expanded with typed features. Some linguistic theories may be directly implemented as a computational formalism. For instance, NIGLE [29, 30] is a well-known programming environment to realize the systemic theory.

In a different social context, NLG systems may cause to generate different text for the same situation. In the systemic theory, it is called *functional variation* of use [30, 37]. A particular functional variety is called **register**. Register, particularly considered in the text planning stage, affects the organization of the grammatical and textual functions, and the choices of lexical items.

In this thesis, we do not considered the details of the text planning because our main purpose is to design and implement a text generator for Turkish. However, the generator requires the semantic description of the text as an input. For that reason, we need to produce the semantic inputs in some way. This thesis assumes that an application (a human in the current system) determines the content and the organization of the text, and then produces the semantic description of the text sentence by sentence. At one time, our text generator takes the semantic description of a sentence, and generates its morphological description that can be worded by the Turkish morphological generator [36]. Consequently, the entire NLG system is organized in the pipelined architecture in which the text planner, an application, produces the semantic description, and then the generator independently realizes it.

In the design, we need to determine *what* kind of semantic description can be given as an input, and *how* it is transformed into an actual text. In order to address these issues, we use a particular linguistic theory known as *Systemic-Functional Grammar* (SFG) that analyzes the semantic and syntactic features of the NL, and the strong relations between them. In the implementation, we use the FUF text generation system and its constraint based formalisms—functional unification and typed features to represent and perform the linguistic resources determined in the design.

Another assumption is that the semantic inputs are also lexicalized according to the register of the text generation system. Thus, the implemented generator can be used as a general syntactic realizer that transforms the semantic description, which is produced and lexicalized by an application, into a real Turkish text.

The remainder of this thesis is organized as follows. In Chapter 2, a brief introduction is given about Systemic Linguistic and its approach to text generation. We consider Turkish Grammar from the Systemic-Functional perspective to describe the linguistic resources in Chapter 3. Next, in Chapter 4, the implementation of Turkish text generator is presented. In Chapter 5, we conclude this thesis and give some directions for the future work. In the Appendix, we present sample runs to demonstrate the generation of the major grammatical units such as clauses, noun groups.

# Chapter 2

# Systemic–Functional Linguistics

Systemic Linguistic has been introduced by M.A.K. Halliday in the early 1960. The origins of systemic linguistics clearly lie in the work of the following major contributers [30, 37]. Bronislaw Malinowski (1884-1942), who was an anthropologist and ethnographers, influenced Firth with the following two ideas: the first one is that language is inseparable from its social and cultural context, and the second one is that language performs certain *functions* in the society, so language is functional. Firth (1890-1960) took and adapted Malinowski's ideas into a linguistic theory with new contributions. Firstly, he introduced the concept of **system**[1] as **a set of linguistic choices in a specific linguistic context**. Then, he considered the differences between "paradigmatic" (system-based) and the "syntagmatic" (structure-based) descriptions of the NL. With these work, Firth created a new linguistic environment which is fundamentally different from the traditional linguistic. The roots of the systemic grammar are described in [41] as follows:

> ... were in anthropology and sociology, not in mathematics or formal logic. The question that motivated its development were not those of grammaticality or the acquisition of linguistic competence, but those of language as a social activity: *What are the social functions of language? How does language fulfill these social functions? How does language work?*

[1]The systemic grammar took its name from the system concept.

The other contribution comes from Hjelmslev (1899-1965). He mainly studied on the *realizational* view of the language. From the realizational perspective, language can be described as a system coded in some level and recoded in another level. Hjelmslev says "semantic, grammar, and phonology are all *semiotics*, or sign systems." Signs at one level can be recoded–or realized–by signs at a lower level (shown in Figure 2.1). Thus, the organizations at different levels are allowed to be independent from one another.

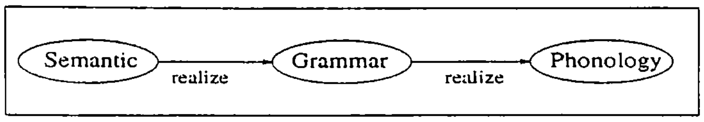

Figure 2.1: Realization of signals at different levels of language

Halliday combined all these work summarized above to form a linguistic theory that is known as *"systemic grammar"*. In the systemic grammar, the linguistic resources are organized as a system network that represents the interrelated choice points in a particular linguistic context [30, 37]. The selected features from the system network enables the related realization rules to explain the meanings with the relevant structures. After this general overview[2] of Systemic–Functional Linguistic, we will try to explain some of the relevant goals of the systemic grammar, and its important concepts. Then, the text generation with this approach is demonstrated, and the well-known software tools are introduced for implementation.

## 2.1 The Goals of Systemic Grammar

The main goals of the systemic grammar can be described as follows [30, 37]:

- To describe the **functions** of language at different levels such as semantic functions–agent, actor, location, and syntactic functions–the "subject" of a clause, the "head" of a noun group and so on.
- To capture the relationships between semantic and syntactic functions [13, 14]. For instance, the semantic function *agent* is mapped onto the syntactic function *subject* in an active clause, and it is realized by a noun group.

[2]More information about systemic linguistic is also available in [30, 37, 41].

- **Classification** of both social meaning and linguistic forms to construct a systemic functional grammar (system–based description). For example, noun groups can be decomposed into three classes: proper, pronoun, and common; common nouns are decomposed into two classes: abstract, concrete and so on. In each class, different meanings or functions are realized by the relevant grammatical structures. In the systemic grammar, the functional and structural descriptions are complementary: the functional description says *"What it does,"* and the structural description says *"How it does it."*

## 2.2 Important Concepts in SFG

In this section, we summarize the main concepts of the systemic grammar [37]. More information can be found in [37, 41].

### 2.2.1 Feature

For the classification of the linguistic resources in the systemic grammar, a feature can be used as the name of a class. For example, some features of a clause (classes in a clause) are *declarative, interrogative, negative, positive* and so on. However, these features are not all independent. For instance, if a clause has the *declarative* feature then it cannot also have *interrogative* one. To represent that kind of mutually exclusive knowledge, the concept of "system" will be considered.

### 2.2.2 System

A system is a mutually exclusive set of classes (or features) and thus represents a choice or "potential" [37]. If a feature is selected in a system, it also means that this system has only that feature, not another one. For instance, if a clause is *positive* then it cannot be *negative*. A particular choice is applicable in some sort of context. For example, to select a feature between declarative and interrogative, the clause must be indicative. In addition, a choice may depend on a logical combination—called "entry conditions" of the system—of more than one selected features (context). The relationships between systems such as entry conditions are represented by drawing "system networks."

### 2.2.3 System Network

System networks display graphically the relationships between features in the grammar [37]. A system represents a choice between two or more features (shown in Figure 2.2). For example, when the system *mood* is entered, one of the features *declarative* or *interrogative* is selected.

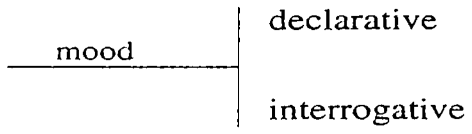

Figure 2.2: System representation

Figure 2.3 illustrates the representation of entry conditions (by drawing lines from the entry condition to the system). For example, the system *mood* can be entered, only if the feature *indicative* is selected earlier.

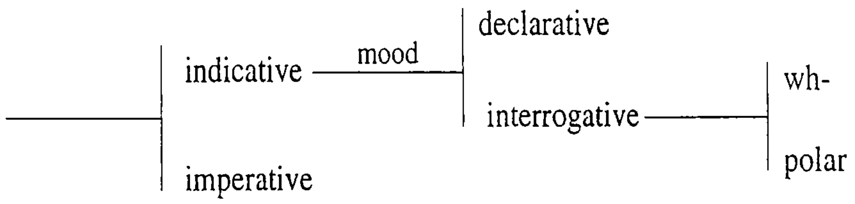

Figure 2.3: The representation of entry conditions

If a feature is an entry condition to more than one system, it is represented by using "{" (it is illustrated in Figure 2.4).

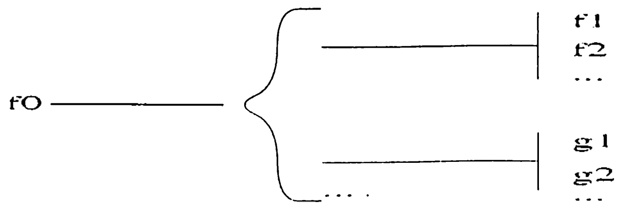

Figure 2.4: Entry condition to several systems

If a particular system has several conjunctive entry conditions (and operation), it is represented by using "}" (it is illustrated in Figure 2.5).

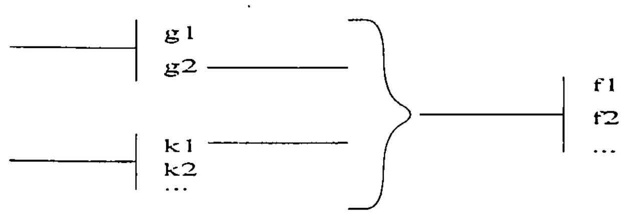

Figure 2.5: Conjunctive entry conditions

To represent disjunctive (not necessarily exclusive) entry conditions (or operation), "]–" is used (shown in Figure 2.6).

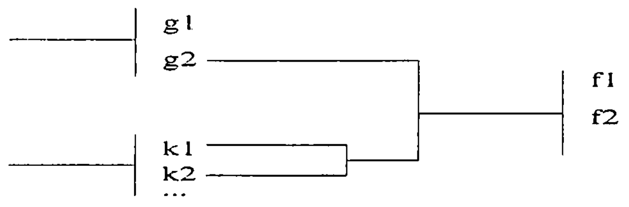

Figure 2.6: Disjunctive entry conditions

There exists a different kind of feature—called "gate"—from the other features in systems above. These features (gates) simply depend on some combination of other features, without choice (shown in Figure 2.7).

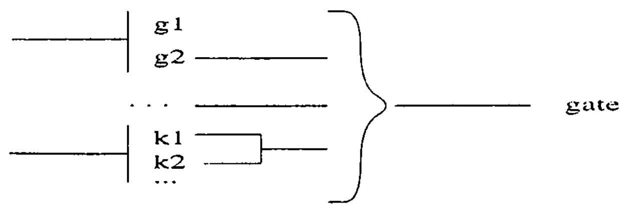

Figure 2.7: A gate from a system network

## 2.2.4 Delicacy

In a classification system, the features of objects are specialized (more information is available about objects) according to previous levels of the classification. In the systemic grammar, this specialization is called *delicacy*. For example, declarative feature is more delicate than indicative feature in Figure 2.3. System networks increase the delicacy from left to right.

### 2.2.5 Functional Analysis

To interpret the meaning of a constituent in a given situation, we use a label that specifies the function of the constituent. In the realization, we use a label that specifies the syntactic class of the constituent. These two types of labels are illustrated in Figure 2.8.

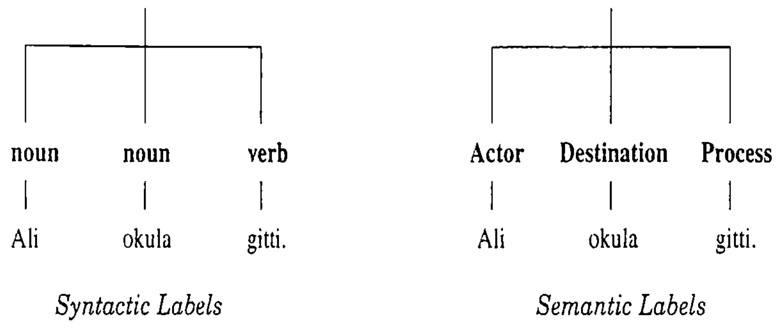

Figure 2.8: Syntactic and Semantic Labels

In the systemic grammar, the linguistic items are analyzed in several functional dimensions simultaneously because a linguistic item may have more than one function at a time. In general, all languages have the following common meta-functional dimensions [14]:

**The IDEATIONAL metafunction** (*Clause as a Representation*) is concerned with ideation: it provides the speaker with the resources for interpreting and representing 'reality' [30]. It represents the logical relationships between processes, events, actions, objects etc.

**The INTERPERSONAL metafunction** (*Clause as an Exchange*) provides the speaker with the resources for creating and maintaining social relations with the listener [30]. It expresses the roles of the speaker in the discourse [37].

**The TEXTUAL metafunction** (*Clause as a Message*) enables the speaker in presenting ideational and interpersonal information as text in context [30]. It ensures that the text is relevant and coherent.

Figure 2.9 shows three functional analysis of the same clause in English. It is taken from [30].

| In this job | Anne | we' | re | working | with silver | |
|---|---|---|---|---|---|---|
| Theme | | Rheme | | | | **textual** |
| | Vocative | Mood | | | | **interpersonal** |
| | | Subject | Finite | | | |
| Locative | | Actor | Process | | Manner | **ideational** |

Figure 2.9: Metafunctional layering in grammar

These metafunctions are common for all languages with some modifications. However, various metafunctions may be required for the functional analysis of different languages. For instance, a different functional analysis is required to deal with the function of free word order in Turkish. Each metafunction consists of more than one functions. For example, the ACTOR, LOCATION, and GOAL functions are used in the analysis of ideational metafunction and so on. We will re-consider these metafunctions and their individual functions for Turkish in Chapter 3.

The related terms of functional analysis are summarized in a tabular form (from [30]) in Figure 2.10.

| | characterization | related typologies | **major resources** |
|---|---|---|---|
| ideational | ideation -- interpretation and representation of the world in and around us | semantic<br>representational<br>denotive<br>propositional context<br>cognive | transitivity (process + participants + circumstances)<br>Actor \| Loc \| Goal \| Proc |
| interpersonal | interaction between speaker and listener; assignment and model-attitudinal comments | conative-expressive (pragmatic) | mood & modality<br>Mood<br>Verb \| Time \| Mode \| per Num |
| textual | presentation of ideational & interpersonal information as text in context; control of textual status and conjunctive development of text | pragmatic<br>discoursal<br>functional sentence perspective | theme and word-order; information; conjunction<br>Theme \| Rheme<br>Topic \| Focus \| verb \| Backg |

Figure 2.10: Metafunctions and Related Terms

### 2.2.6 Rank

Although the systemic grammar deals with the functional issues of language, it must still relate the function to structure. The linguistic items (constituents) are grouped together in a separate level of structure. The hierarchical relationships between the various units is called RANK–from "largest" to "smaller." Figure 2.11 shows the rank system in Turkish grammar.

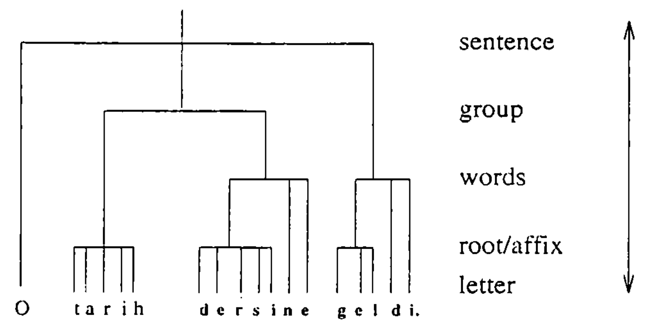

Figure 2.11: Rank in Turkish grammar

### 2.2.7 Realization Rules

"Realization rules" are used to construct the relationships between the features and system networks on one hand, and the functional analysis and constituent structure on the other. In this way, the structural representation is also achieved. The elements of structure are represented in realization rules by their function (e.g., Agent, Actor). The realization relationships between linguistic items vary from language to language. For example, Turkish is a free word order language, so there is no strict order between the constituents (e.g., Actor and Process may or may not be adjacent). However, the order of constituents in English is not free (e.g., Actor and Process must be adjacent). The following operators can be used in the realization process.

/ **Conflation** : to specify the identity of two functions (e.g. Agent/Subject).

+ **Insertion** : to insert a new function (e.g. +Subject).

: **Preselection** : to select a feature before it is actually encountered (e.g Subject:noun-group). It takes place from one rank to the next rank below.

:: **Lexical Preselection** : to select a lexical item to realize the related function (AgentMarker::`tarafından` (by)).

O **Expansion** : to divide a function into sub-functions (e.g. Mood(Subject) and Mood(Finite)).

∧ **Order** : to realize linguistic items as adjacent in the structure (e.g. Focus ∧ Process or Subject ∧ # or # ∧ Process). # is used to represent a leftmost or rightmost constituent.

··· **Order** : to represent partial orderings of the linguistic items in the structure (e.g. Subject ··· Object ··· Process).

These operators are used to describe the realization rules in the system network. Their implementation depends on the computational formalism.

## 2.3 Systemic–Functional Text Generation

According to SF approach, the linguistic resources are organized into a number of levels for making and expressing meanings as shown in Figure 2.12. These

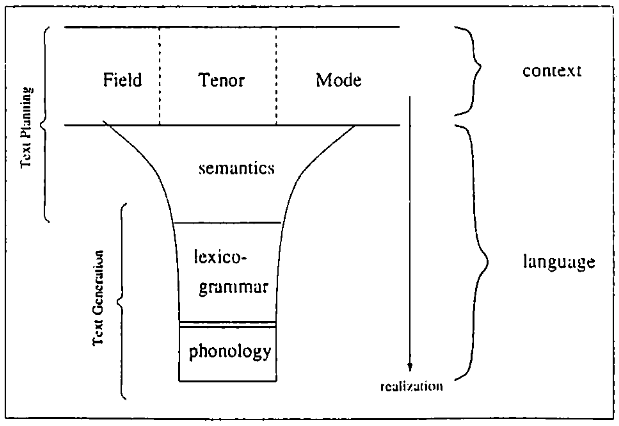

Figure 2.12: Language as a tristratal system

levels, called STRATA in systemic terminology, represent the different order of abstraction in NLG. From top end to the bottom end, a higher strata is realized into a lower one and so on to accomplish the NLG. The context and semantic levels are considered in text planning stage [27]. The text generation stage deals with the outputs of semantic strata, and the lexico-grammar strata. Here, we will consider the text generation rather than planning. Hence, we need

- to determine the semantic inputs, and
- to organize the linguistic resources at lexico–grammar strata that represents how to transform the semantic description of the text into an actual text [28].

In the systemic–functional approach, functional analysis of NL gives us the potential semantic descriptions, and the SFG describes the correspondence between the meanings and their NL expressions by organizing the linguistic resources at lexico–grammar strata. Now, we want to exemplify the generation of a simple sentence given in (1) to demonstrate the SF approach.

| | | | |
|---|---|---|---|
| (1) | Ali | camı | kırdı. |
| | Ali | window+3SG+ACC | break+PAST+3SG |
| | 'Ali broke the window.' | | |

The semantic description of this sentence can be given by using the following functions:

**Process** `(kırdı)`: the performed action

**Actor** `(Ali)`: the first participant that does the deed

**Goal** `(camı)`: the second participant that suffers the process

**Agent** `(Ali)`: the causer of the process

**Medium** `(camı)`: the affected constituent from the process

This sentence will be realized in active voice. Table 2.1 gives the multidimensional functional representation of this sentence. The required part of SFG is presented in Figure 2.13.

In text generation with SFG, the system network is traversed from left to right by following the basic sequential traversal algorithm as shown in Figure 2.14. This algorithm is originally presented in the NIGLE which has several distinct activities such as Environment, Choosers, Grammar and Realizer [29, 30]. They will be described in Section 2.4. Here, we assume that Choosers ask some questions to the Environment and the Environment gives appropriate answers. Thus, the features are selected by the Chooser, and then

| Ali | camı | kırdı. | *Func. Dimensions* |
|---|---|---|---|
| Agent | Medium | Process | **Ideational** |
| Actor | Goal | Process | |
| | | Voice,Mood | **Interpersonal** |
| Topic | Focus | Verb | **Textual** |
| Subject | D.Obj | Predicate | **Syntactic** |

Table 2.1: Multidimensional Functional Analysis

the Grammar and the Realizer realizes the functions which are attached to selected features.

If we execute this algorithm manually on the system network given in Figure 2.13, the following systems are entered and the appropriate features are selected: Enter Rank system, select sentence; enter ProcessType, select material; enter Agentive, select yes; enter Effective select yes; enter Voice, select active; directly enter G2 gate. If we assume that the semantic roles are lexicalized, and realization rules attached to each selected feature are executed, then the semantic roles are mapped onto syntactic roles, and ordered as shown in Table 2.1. As a result, the generation of the sentence given in (1) is completed.

## 2.4 Software Tools for Implementation

In the implementation, different approaches may be used to represent the SFG on computer, and to provide communication between semantic strata and lexico-grammar strata. Here, we will introduce four different software environments: FUF, GENESYS, WAG, and NIGLE.

FUF is a general purpose text generation system that uses the constraint based formalisms—functional unification grammar (FUG) techniques and typed features [7, 8]. We use this generation system in our implementation. So, in Chapter 4, FUF and its approach to the implementation of a text generator will be considered in more detail.

GENESYS provides an integrated environment for developing SFGs [26]. GENESYS also uses the FUG formalism like FUF. In addition, a graphical user interface (GUI) is included to the environment for editing and processing

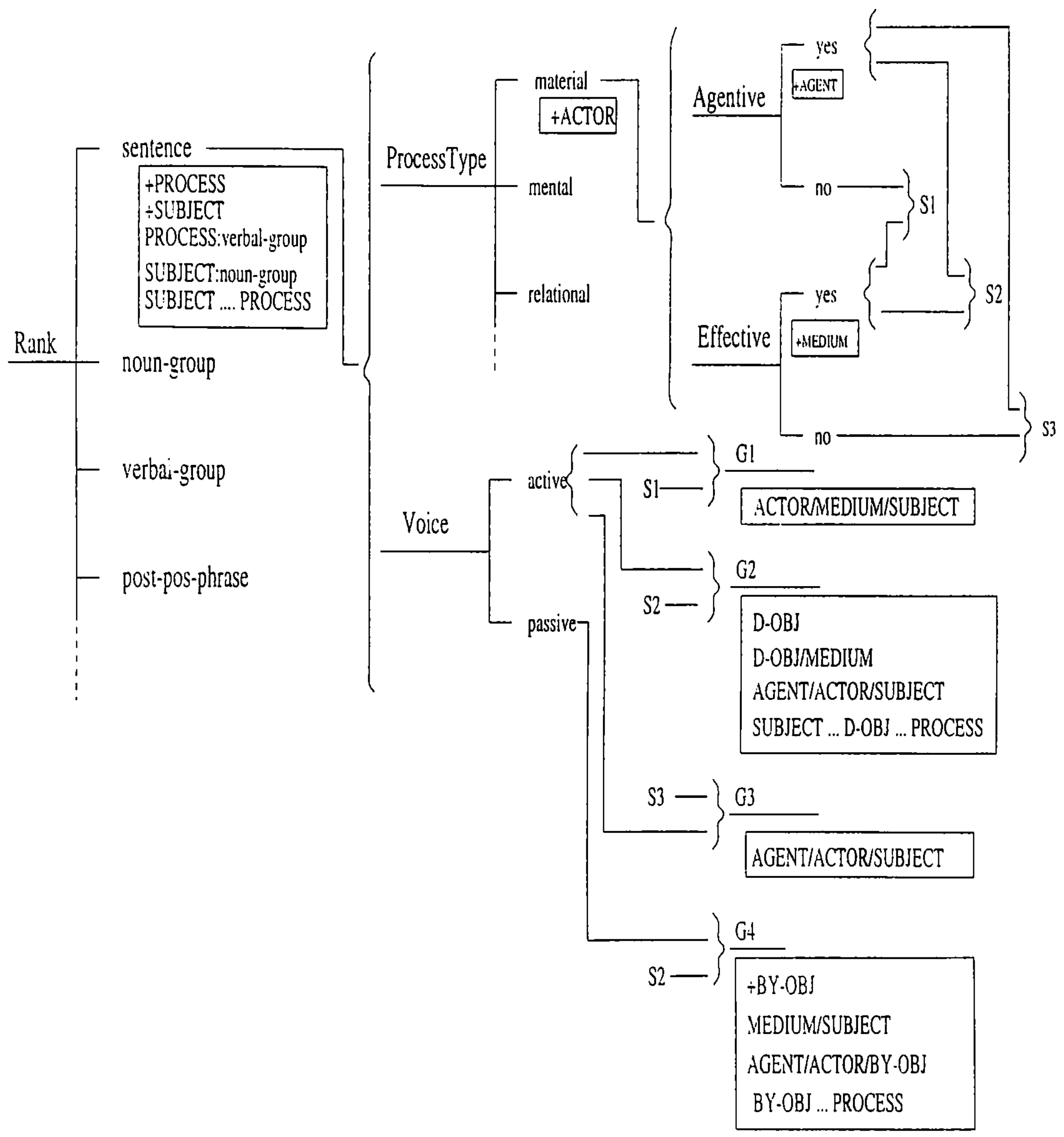

Figure 2.13: A Partial System Network for the Sentence Generation

the linguistic resources.

WAG Sentence Generation System is one of the Workbench for Analysis and Generation (WAG) that provides several tools to represent and perform the systemic resources [34, 35].

NIGLE is the first implemented systemic grammar for text generation [29]. It is a part of larger text generation system called Penman [30]. NIGLE can be characterized by having the following distinct activities:

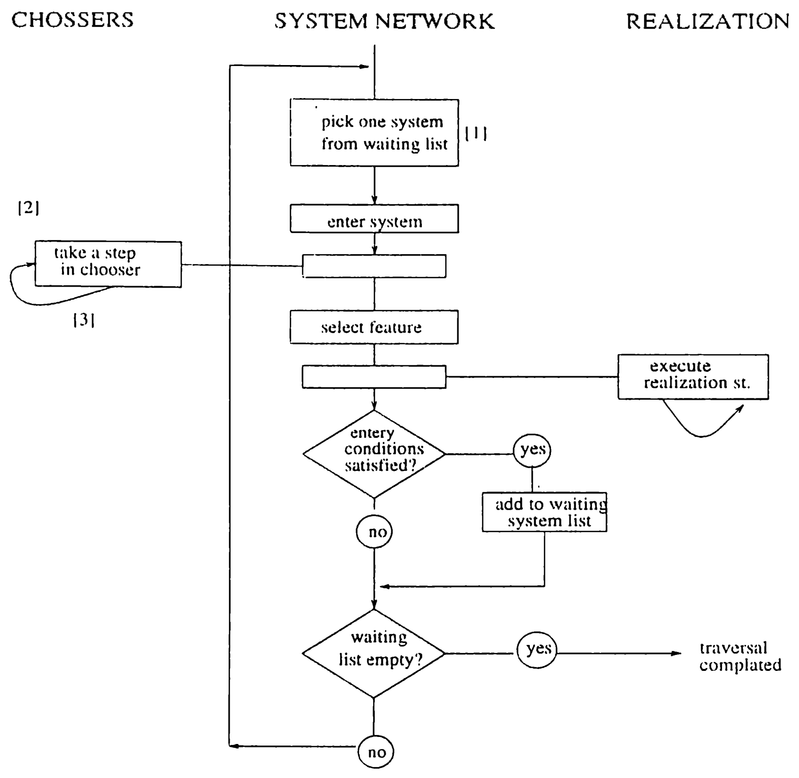

Figure 2.14: A basic sequential traversal algorithm

**Environment:** contains the representation of three kinds of knowledge ( Knowledge Base, Text Plan and Text Service).

**Choosers:** Each system has a chooser. When the grammar enters a system, its chooser is activated. Then, it asks the Environment a question to select a feature in the current system.

**Grammar:** contains the systems of the whole systemic grammar. It enters systems and keeps track of the selected features.

**Realizer:** It shows each realization as soon as it becomes definite.

In the generation, the systemic grammar network is traversed from left to right by selecting relevant features and executing realization rules according to the responses of environment to the chooser questions. The basic sequential traversal algorithm of NIGLE has already been presented in Figure 2.14.

# Chapter 3

# Turkish Grammar

## 3.1 Sentence

Sentence is a sequence of words that forms a statement, command, exclamation, or question. These forms of sentences completely explain the ideas, orders, sudden strong feelings, and demanded information, respectively. In writing, sentence begins with a capital letter and ends with one of the punctuation marks ". ! ?" according to its form. To generate a sentence, at least, a *Subject* and a finite verb called *Predicate* are required. If the Subject is a pronoun, it may be omitted because a Predicate in Turkish already contains the person information of Subject. In that case, Predicate, a finite verb, becomes a sentence that owns only one word. This single word sentence may be called "Core Sentence" as shown in (2).

(2) a. Gitti.
go+PAST+3SG
'(He) went.'

b. Zordur.
difficult+COP+AOR+3SG
'(It) is difficult.'

The core sentence can be extended by using extra elements to explain the additional information in the sentence. The following sentences give more information about the main process in (3.a) by extending it with new elements:

(3) a. Kırdı.
break+PAST+3SG
'(He) broke.'

b. Ali kırdı.
Ali break+PAST+3SG
'Ali broke.'

c. Ali camı kırdı.
Ali window+ACC break+PAST+3SG
'Ali broke the window.'

d. Ali camı dün kırdı.
Ali window+ACC yesterday break+PAST+3SG
'Ali broke the window yesterday.'

Sentence is not only a sequence of words but also a semantic representation of reality. In fact, the sentence can be represented in more than one level:

| | |
|---|---|
| Phonology | sound |
| Orthography | writing |
| Syntactic | wording |
| Semantic | meaning |

But we deal with only the last two levels. In these two levels, we use *Syntactic* and *Semantic* functions to represent the sentences. The semantic functions represent the meaning (semantic role) of the elements in the sentence as shown in Table 3.1. The syntactic functions represent the grammatical role of these

| Ali | camı | kırdı |
|---|---|---|
| Agent | Medium | Process |
| Actor | Goal | Process |

Table 3.1: Representation of Sentence with Semantic Functions

elements as shown in Table 3.2.

| Ali | camı | kırdı |
|---|---|---|
| Subject | D. Object | Predicate |

Table 3.2: Representation of Sentence with Syntactic Functions

Each language has an individual lexicogrammar—lexicon and grammar that provides a way to unify the semantic functions with the syntactic functions. For example, the general semantic functions can be unified with the following syntactic functions.

| *Semantic Functions* | *Syntactic Functions* |
|---|---|
| Process | Predicate |
| Participants | Subject & Objects |
| Circumstances | Adjuncts |

In this sense, the relation between grammar and semantic is natural, not arbitrary. This relation, that pushes the grammar into the semantic level, is especially considered in the *Functional Grammar* [14]. In addition, the grammar encodes the unified semantic and syntactic functions as a worded text by using the *syntactic structures*. This process is called *realization*. For example, the unified syntactic and semantic functions can be realized as follows.

| *Semantic Functions* | *Syntactic Functions* | *Syntactic Structure* |
|---|---|---|
| Process | Predicate | verbal group |
| Participants | Subject & Objects | noun group (NP) |
| Circumstances | Adjuncts | NP, PP, AdvG |

Thus, the relation between syntax and grammar is natural too. In that case, the relation pushes the grammar into the syntactic (structure-based) level. To describe and perform the linguistic resources at those two levels, systemic-grammar provides us with a reasonable approach that was introduced in Chapter 2.

In the implementation, we will use the term "clause" rather than sentence. A clause may be described as a configuration of participants and circumstantial functions around a central process. It is also a common name for sentence and sentence-like structures. In the systemic grammar, the *mood* system (shown in Figure 3.1) determines the usage form of the clause. At the top level, mood

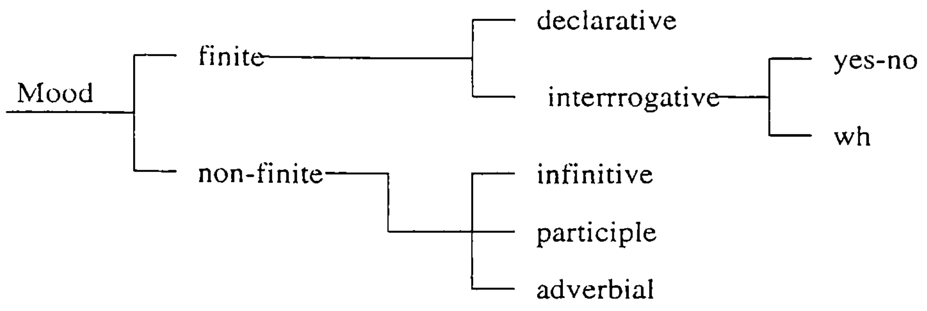

Figure 3.1: Mood System Network

system presents two alternatives: finite, and non-finite. Finite clauses are used as the simple sentences considered in Section 3.2.2. Non-finite clauses are used as noun, adjective, or adverb in other grammatical units. Their computational generation will be discussed in Chapter 4.

## 3.2 Classification of Sentences

In the grammar, Turkish sentences are divided into more than one classes according to their:

- forms
- structures
- predicate types
- word orders

Each of these classes will be discussed in the following sub-sections.

### 3.2.1 Sentence Forms

Sentence generation is an interactive event involving a *Speaker or Writer* and a *Listener or Reader* [14]. Speaker generates a sentence to exchange *information*, or *goods-&-services* with Listener. The role of Speaker and Listener in exchange may be seeker and supplier or supplier and seeker, respectively. Two speech roles (giving & demanding) and the characteristics of the exchanged commodity determine the sentence form. According to these criteria, the following sentences can be classified as shown in Table 3.3.

(4) a.

| | | |
|---|---|---|
| Çay | ister | misiniz? |
| tea | want+AOR | Ques+2PL |

'Would you like tea?'

b.

| | | |
|---|---|---|
| Ali | okula | gitti. |
| Ali | school+DAT | go+PAST+3SG |

'Ali went to school.'

c.

| | |
|---|---|
| A | kırıldı! |
| A | break+PASS+PAST+3SG |

'It was broken!'

d.

| | |
|---|---|
| Kapıyı | aç. |
| door+ACC | open+IMP |

'Open the door.'

e.

| | | | |
|---|---|---|---|
| Ali | okula | gitti | mi? |
| Ali | school+DAT | go+PAST+3SG | Ques |

'Did Ali go to school?'

| commodity exchange | goods-&-services | information | strong feelings |
|---|---|---|---|
| role in exchange | | | |
| giving | *offer as question*<br>Çay ister misiniz? | *statement*<br>Ali okula gitti. | *exclamation*<br>A kırıldı! |
| demanding | *command*<br>Kapıyı aç. | *question*<br>Ali okula gitti mi? | —<br>— |

Table 3.3: Sentence Forms According to Speech Role and Commodity Exchange

To determine the sentence form with the terms of systemic grammar, the required two systems (`Speech-Role`, `Commodity-Exchange`), and the relations between them are graphically represented in Figure 3.2. All forms of sentences may be positive or negative. So, the polarity of a sentence is determined by a separate system.

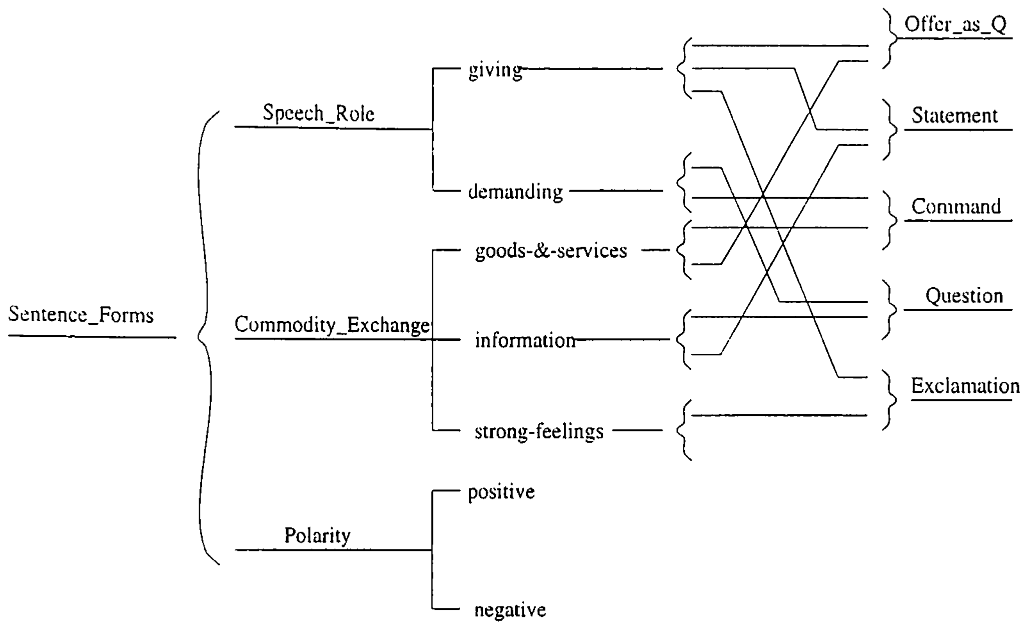

Figure 3.2: A System Network for Sentence Forms

### 3.2.2 Sentence Structures

In the traditional grammar, Turkish sentences are structurally divided into four main classes: *simple, compound, connected,* and *coordinate* sentences [2, 19, 24]. It is not possible to say that there is a sharp line between these classes. For instance, simple and compound sentences have the same functional constituents but they are assumed in distinct classes because of the different realizations of some constituents in the sentence. From the functional view point, the structure of a sentence can be determined by the configuration of its functional constituents without considering their realizations. In this context, according to the configuration of constituents, the sentences can be decomposed into two classes: *simple* and *complex.*

*Simple Sentence* consists of only one main process and several components that complement or modify the main process. Each component may be realized by complex syntactic structures but it does not change the simple structure of sentence. In other words, the number of words in a sentence does not determine whether the sentence is simple or not. The main property of the simple sentence is that each component in the sentence has a function that is determined by the main process such as time, location, actor, reason, manner etc.[1] The traditional simple and compound clauses are considered as simple sentences. The simple sentences can be exemplified as follows (in traditional grammar, (5.a) and (5.c) are called simple and compound, respectively).

(5) a.

| | |
|---|---|
| Ali okula | gitti. |
| Ali school+DAT | go+PAST+3SG |

'Ali went to school.'

b.

| | | |
|---|---|---|
| *Okula* | *giden* | *adam* |
| camı | kırdı. | |
| school+DAT | go+CONV=ADJ | man |
| window+ACC | break+PAST+3SG | |

'*The man who went to school* broke the window.'

c.

| | | |
|---|---|---|
| Adam | *okula* | *giderek* |
| camı | kırdı. | |
| man | school+DAT | go+CONV=ADV |
| window+ACC | break+PAST+3SG | |

'The man broke the window by going to school'

*Complex Sentence* consists of more than one simple sentence that may be structurally (6.a) or semantically (6.b) connected to each other [19].

(6) a. Hafta sonları, kütüphaneye gider(dik) ve kitap okurduk.
'At the weekends, we used to go to library and read the book.'

b. Çok yorgun olmasına rağmen Ali işe gitti.
'Although he was very tired, Ali went to work.'

We assume that the the traditional connected and coordinated sentences are *complex*. However, the complex sentences are not considered in more detail. Because the simple sentence generation must be achieved before generating a complex sentence. For that reason, the main focus of this thesis will be on the simple sentences.

[1] See Section 3.3 for more information about the functional constituents in the simple sentence.

### 3.2.3 Predicate Type of Sentences

In Turkish grammar, sentences can be divided into two groups according to the type of their predicates: *verbal*, and *nominal* sentences. If the predicate of the sentence is derived from a verb, it is called a verbal sentence. It can be exemplified as follows.

(7) a. Yarın sabah okula *gideceğiz.*
Tomorrow morning school+DAT go+FUT+1PL
'We will go to school tomorrow morning.'

b. Çocuklar top oynamayı *severler.*
child+3PL football play+CONV=NOUN+ACC like+AOR+3PL
'The children like playing football.'

If the predicate is derived from a nominal[2] group, it is called a nominal sentence. In the realization of the nominal sentences, the following exceptions must be considered: A nominal group becomes a finite verb with a *substantive (predicative)* verb that is used as an auxiliary verb (copula) to demonstrate the "to be" meaning of the predicate in the following four *grammatical tenses*: *Aorist*, *Past*, *Narr*, and *Cond*.

(8) a. Kız çok *güzeldi.*
girl very beautiful+COP+PAST+3SG
'The girl was very beautiful.'

b. Ahmet *başkandır.*
Ahmet chairman+COP+AOR+3SG
'Ahmet is the chairman.'

The negative sense of the nominal sentence is represented by a separate word "`değil`" (not to be) for the mentioned four tenses above.

(9) Ahmet *başkan değildir.*
Ahmet chairman NegNoun+COP+AOR+3SG
'Ahmet is not the chairman.'

[2]Nominal is a common name for noun and adjective.

To explain the other tenses (*future, progress, optative, necessitative, imperative*) in the nominal sentences, the auxiliary verb "`olmak`" (to be) is used. In that case, the auxiliary verb is realized as the predicate of a verbal sentence.

(10) a. Yarın *okulda* *olacaksın.*
tomorrow school+LOC be+FUT+2SG
'You will be at school tomorrow.'

a. Ahmet *başkan* *olmayacak.*
Ahmet chairman be+NEG+FUT+3SG
'Ahmet will not be the chairman.'

One of the most used nominal sentences is the existential sentence "var, yok" (existent, absent). There is no different issue in the realization of existential sentences. The process of an existential clause is derived from the noun `var` or `yok` to express that something exists or not.

(11) a. Masada üç kitap *vardı.*
table+LOC three book exist+COP+PAST+3SG
'There were three books on the table.'

b. Masada hiç kitap *yoktu.*
table+LOC any book absent+COP+PAST+3SG
'There wasn't any book on the table.'

This syntactic classification does not provide any more information on how to determine the constituents of a sentence. In the functional analysis of Turkish (in Section 3.3), we divide sentences into several groups according to the meaning of their process to determine the relevant sentence configurations.

### 3.2.4 Word-Order in the Sentence

Turkish is a free word order language, for instance the syntactic functions can be ordered as follows:

- Regular Sentence: *Subject* ··· *Object* ··· *Verb* (default)

- Irregular Sentence $V \cdots S \cdots O$ or $O \cdots V \cdots S$

Although the same constituents are freely ordered to construct a sentence, each order provides the additional information to explain the different textual function of each constituent. The textual functions can be identified as follows [10, 15]:

- the sentence-initial position as *topic*
- the immediately preverbal position as *focus*
- the postverbal position as *background information*

In the realization, each constituent may be conflated with one of these functions, and these functions are strictly ordered as shown in the following template:

$$\underbrace{Topic\ Neutral}_{Theme} \cdots \underbrace{Focus\ Process\ Background}_{Rheme}$$

Naturally, the number of constituents in the sentence may be increased, and they can not be conflated any textual function. For those kinds of constituents, we will use a default word order in the implementation (see Section 4.2). Actually, it is not possible to generate a natural sentence in this way. To solve this problem, we need more linguistic analysis.

In spite of the free word order characteristic of Turkish. there are some grammatical constraints on the word order. If Direct Object is not focused in the sentence (12.a), it must be realized as a definite element. If Direct Object is an indefinite element (12.b), it must be adjacent with the process. Otherwise, it will be ungrammatical (12.c).

(12) a. *Camı* Ali kırdı.
window+ACC Ali break+PAST+3SG
'Ali broke the window.'

b. Ali *cam* kırdı.
Ali window+NOM break+PAST+3SG
'Ali broke (a) window.'

c. * *Cam* Ali kırdı.
window+NOM Ali break+PAST+3SG

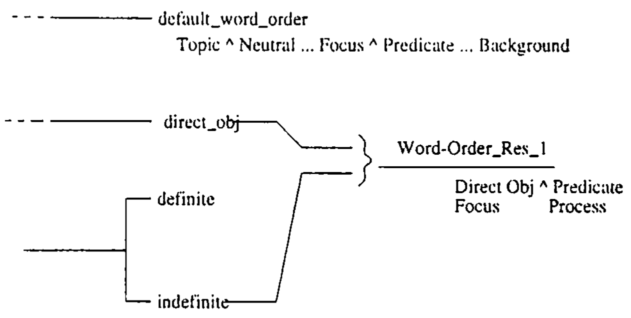

Figure 3.3: A System Network for Word-Order

Figure 3.3 shows a system network that can be used to realize a part of word-order restrictions in Turkish.

## 3.3 Functional Analysis

From the functional perspective, all languages try to realize the common semantic functions with their own grammatical structures and lexical items. According to Halliday, all languages have the following three common metafunctions:

- Ideational
- Interpersonal
- Textual

These metafunctions had already been described in Section 2.2.5. Here, we will use more specific functions given by Halliday for each metafunction to describe the semantic configuration of a clause.[3] Then, we will consider the realization of each semantic function in Turkish.

[3]Clause is used as a common name for sentence, or sentence-like structure. It can be described as a configuration of participants and circumstantials around a central process.

### 3.3.1 Ideational Representation

Ideational representation of a clause consists of three functional components: *process*, *participants*, and *circumstantials*. **Process** is the main constituent that represents an *event* or a *state*. **Participants** are persons or things involved in a process. **Circumstantials** are the optional constituents to describe the process from different perspective such as time, place, manner etc.

Participants, and Circumstantials are specified with new semantic functions to represent the special meanings, roles or relations in the clause. The specific participant functions depend on the type of process. The *transitivity* and *ergativity* analysis [14] allow us to classify the processes in the language, and to describe the configuration of participants. The specific circumstantial functions do not strictly depend on the type of the process. They are optionally used to give more information about the process.

In the following sub-sections, we will present the transitivity and ergativity analysis for Turkish. Then, we will consider the realization of participants and circumstantials. By the way, the process of a clause is realized by a verbal group presented in Section 3.4.

#### Transitivity

Transitivity specifies the different types of processes recognized in the language, and determines the participants according to these types. In this way, the logical relationships between the process and participants are provided. The types of processes and their special participants may be classified as follows.

1. *Material processes (processes of doing)* express the notion that some entity "does" something which may be done "to" some other entity [14]. That kind of process contains the following two participants (also shown in Table 3.4): **Actor** is an obligatory participant that represents the one that does the deed. **Goal** is an optional participant that represents the one that the process is extended to. Another term that may be used for this function is *patient*.

   The material processes are characterized by the following two semantic features: *agentive* and *effective*. Each of these features may be *yes* or *no*

| Ali | camı | kırdı |
|---|---|---|
| Actor | Goal | Process |

Table 3.4: Involved Participants in Material Processes

but both of them can not be *no* at the same time. Thus, three different alternatives appear as shown in Figure 3.4. At each alternative, a distinct

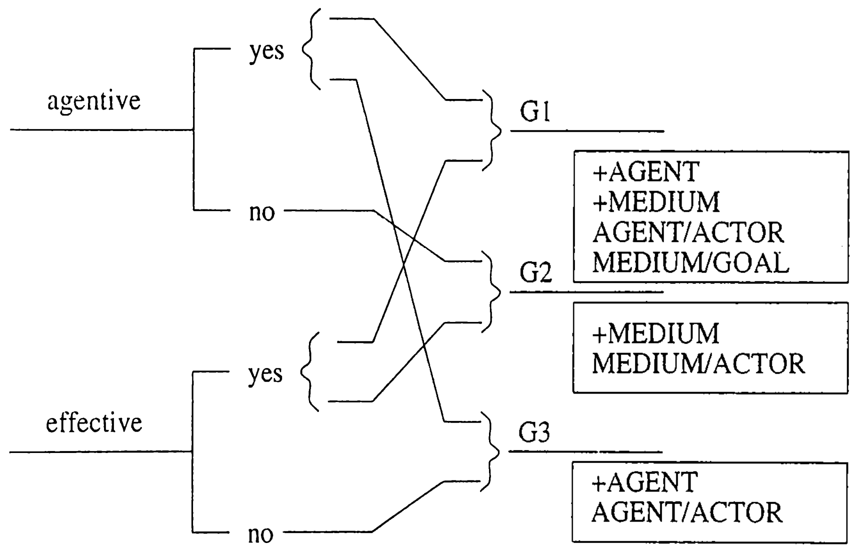

Figure 3.4: Semantic features of the material process

configuration of the participants is used for the realization. The participants *agent* and *medium* will be described in the ergativity analysis (next section).

2. *Mental processes (processes of sensing)* express feeling, thinking, and perceiving activities of humans. There are two participants in a mental process (also illustrated in Table 3.5): **Senser** is the conscious being that feels, thinks or senses. **Phenomenon** is a thing or a fact that is "sensed"-felt, thought or seen.

| | | | |
|---|---|---|---|
| (13) | Çocuklar | kitap okumayı | severler. |
| | child+3PL | book read+CONV=NOUN+ACC | like+AOR+3PL |
| | 'The children like reading book.' | | |

Mental processes can be divided into three sub-types [14]: Perception (seeing, hearing etc.), Affection (liking, fearing etc.), Cognition (thinking,

| Çocuklar | kitap okumayı | severler. |
|---|---|---|
| Senser | Phenomenon | Process |

Table 3.5: Involved Participants in Mental Processes

knowing, understanding etc.). All mental processes potentially involve both a senser and a phenomenon.

3. *Relational processes (processes of being)* express the way of "being." The central meaning of sentence in this type is that *something is.* The relational processes can be classified according to the type of *being*, and the explanation mode of *being*. The type of being may be one of the followings [14]:

| | | |
|---|---|---|
| (1) Intensive | "$x$ is $a$" | $x$ $a$`-dır` |
| (2) Circumstantial | "$x$ is at $a$" | $x$ $a$`-da-dır` |
| (3) Possessive | "$x$ has $a$" | $x$ $a$`-ya sahip-tir` |

Each type can be explained in two modes [14]:

| | |
|---|---|
| (a) attributive | "$a$ is an attribute of $x$" |
| (b) identifying | "$a$ is an identity of $x$" |

As a result, the six types of relational processes can occur. Each type of relational processes may be exemplified as follows. They are also classified in Table 3.6 according to the mode and type of being.

(14) a. Bu kitap siyahtır.
this book black+COP+AOR+3SG
'This book is black.'

b. Ali okuldadır.
Ali school+LOC+COP+AOR+3SG
'Ali is at the school.'

c. Ali üç kitaba sahiptir.
Ali three book+DAT own+COP+AOR+3SG
'Ali has three books.'

(15) a. Ali başkandır.
Ali chairman+COP+AOR+3SG
'Ali is the chairman.'

b. En iyi yer okuldur.
best place school+COP+AOR+3SG
'The best place is the school.'

c. Ali'nin kitabıdır .
Ali+GEN book+POSS=3SG+COP+AOR+3SG
'It is Ali's book.'

| *mode of being* / type of being | *attributive* | *identifying* |
|---|---|---|
| Intensive | Bu kitap siyahtır. | Ali başkandır.<br>Başkan Ali'dir. |
| Circumstantial | Ali okuldadır. | En iyi yer okuldur.<br>Okul en iyi yerdir. |
| Possessive | Ali üç kitaba sahiptir. | Ali'nin kitabıdır.<br>Kitap Ali'nindir. |

Table 3.6: Relational Processes

The special participants for each type of relational processes are determined according to the mode of *being*. In the attributive mode (illustrated in Table 3.7), an attribute is ascribed to some entity [14]. Participants in this mode are as follows: **Carrier** is an entity to that an attribute is ascribed. **Attribute** is a determiner that is ascribed to Carrier. In the

| *attribute of* | | | |
|---|---|---|---|
| Intensive | Bu kitap | siyah | -tır |
| Circumstantial | Ali | okul-da | -dır |
| Possessive | Ali | üç kitaba | sahiptir |
| Participants | Carrier | Attribute | Process |

Table 3.7: Attributive Mode in Relational Processes

identifying mode (illustrated in Table 3.8), one entity is used to identify another. To represent these two entities, the participants **Identifier** and

| | | |
|---|---|---|
| Masanın üzerinde | üç kitap | vardı. |
| Masanın üzerinde | hiç kitap | yoktu. |
| Location | Entity | Process:(existential yes) |
| Location | Entity | Process:(existential no) |

Table 3.9: Participant in Existential Process

5. *Other types of process* may be recognized in the language such as verbal, behavioral [14]. In this study, we do not deal with these kinds of processes.

## Ergativity

If the process is "caused"-ergative, the analysis of the ergativity is required to find the following functions as participants: **Agent** (the Causer), **Medium** (the Affected). Sometimes Medium is conflated with Actor, sometimes with

| | | |
|---|---|---|
| Ali | camı | kırdı |
| Agent | Medium | Process |
| Actor | Goal | Process |

| | |
|---|---|
| Cam | kırıldı |
| Medium | Process |
| Actor | Process |

Table 3.10: Participant Configuration with Ergativity Analysis

Goal as shown in Table 3.10. This multidimensional functional treatment and the interaction of functions are the key points in the functional approach to generate a text.

In addition, the agent and the actor may be different participants to explain the fact that someone (agent) is causing someone else (actor) to perform the process. For instance, in (17), `Ali` (the agent) was causing `Veli` (the actor) to paint the table.

| | | | | |
|---|---|---|---|---|
| (17) | Ali masayı | Veli'ye | boyattı. | |
| | Ali table+ACC | Veli+DAT | paint+CAUS+PAST+3SG | |

'Ali had Veli paint the table.'
'Ali had the table painted by Veli.'

In Turkish, the causation hierarchy may be more complex. More than one agent-like participants may appear between the agent and the actor to explain that someone is causing another causing another and so on to perform the process. For example, an additional participant (Agent-2) is illustrated in (18).

(18) Ali masayı Ahmet aracılığı ile Veli'ye boyattırdı.
Ali table+ACC Ahmet's help with Veli+DAT paint+CAUS+CAUS+PAST+3SG
'Ali told Ahmet to have Veli paint the table.'

We will not consider more complex causations, because they are not frequently used in practice.

## Common Participants

Some participants can occur in more than one type of processes. **Beneficiary** (*logical indirect object*) is the one to whom or for whom the process is done. It may occur in: Material processes and Relational processes. **Range** (*logical cognate object*) is a component that represent the range or scope of the process. It may appear in: Material processes and Mental processes. Those two participants can be exemplified respectively as follows.

(19) a. Ali *Veli'ye* bir kitap verdi. (Beneficiary)
Ali Veli+DAT a book give+PAST+3SG
'Ali gave Veli a book.'

b. Ali *tenis* oynuyor. (Range)
Ali tennis+NOM play+PROG+3SG
'Ali is playing tennis.'

## Realization of Participants

Participants are mapped onto syntactic functions such as *subject*, *direct-object etc.*, and generally realized by noun groups, and infinitive clauses. In this study, the semantic functions given in Table 3.11 are used to represent the participants in the different types of processes.

| Process Type | Semantic Functions |
|---|---|
| Material | actor, goal, agent, medium |
| | beneficiary, range |
| Mental | senser, phenomenon, agent |
| | beneficiary, range |
| Relational - Attributive | carrier, attribute |
| Relational -- Identifying | identified, identifier |
| Existential | entity |
| Others - Verbal, Behavioral | *are not considered* |

Table 3.11: Participant Configuration in Simple Processes

## Realization of Circumstantials

In contrast to participants, circumstantials are not mapped onto any syntactic functions. They are directly realized by noun groups, post-positional groups or adverbs in the sentence structure. In this approach, there is no syntactic distinction among several circumstantials. However, Quirk decomposed circumstantials (that they call adverbial) into different classes (as shown in Figure 3.5 from [38]) according to their syntactic behavior.

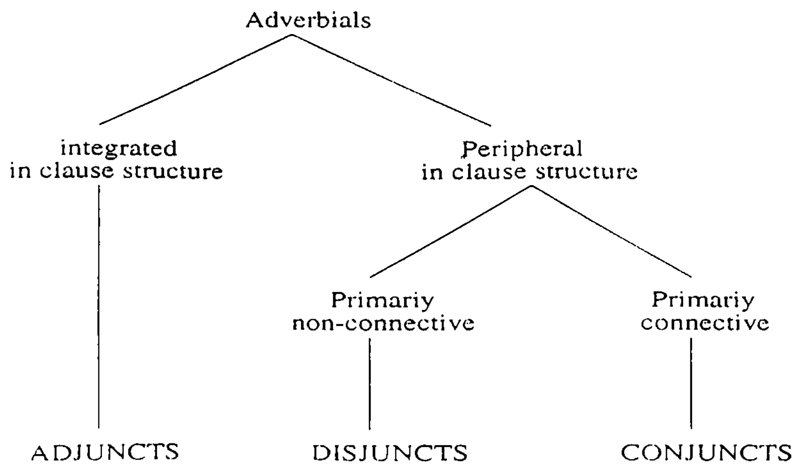

Figure 3.5: Adverbials (Syntactic Classification of Circumstantials)

In this study, we have considered only the circumstantials that are presented in [14], and realized by adjuncts [38]. Adjuncts may be one of the following grammatical structures: Noun Group (NP), Post-Positional group (PP), Adverb Group (AdvG). More information about these units can be found in Sections 3.5, 3.6, and 3.7, respectively. The disjuncts and conjuncts are presented in [38], and most of them are implemented in SURGE 2.0 (see [9, 17]).

Circumstantial functions can be decomposed into seven classes (as shown in Table 3.12) according to their functional features. The number of these classes may increase by depending on linguistic analysis.

| Class | Semantic Func. | Answer the Ques. | Realizations |
|---|---|---|---|
| Spatial | direction | in what direction? | AdvG, PP, NP |
| | distance | how far? | NP |
| | origin | from where? | NP |
| | location | where? | NP |
| | destination | to where? | NP |
| | path | through where? | NP, PP |
| Temporal | duration | how long? | NP, AdvG, PP |
| | frequency | how often? | AdvG, PP |
| | time | when? | NP, AdvG, PP |
| Manner | means/instrument | how?/what with? | NP, PP |
| | quality | how ...? | NP, AdvG |
| | comparison | what like? | PP |
| Cause | reason | why? | PP |
| | purpose | what for? | PP |
| | behalf | who for? | PP |
| Accom- | comitative + | what/who with/else? | PP |
| -paniment | comitative − | but not who/what? | NP |
| | additive + | and who/what else? | PP |
| | additive − | and not who/want? | PP |
| | Matter | what about? | PP |
| | Role | what as? | PP |

Table 3.12: Circumstantials Realized by Adjuncts

Each circumstantial function given in Table 3.12 can be described, and exemplified according to its syntactic realizations as follows.

1. *Extent-&-Location in space* (Spatial Functions)

    - **Direction** is a function that represents the direction of the process.

(20) a. Ali *aşağıya* koşuyor. (AdvG)
Ali down run+PROG+3SG
'Ali is running down.'

b. Ali *okula* *doğru* koşuyor. (PP)
Ali school+DAT towards run+PROG+3SG
'Ali is running towards the school.'

c. Ali *ters* *yön(d)e* koşuyor. (NP)
Ali opposite direction+DAT/LOC run+PROG+3SG
'Ali is running at the opposite direction.'

- **Distance** is a function that represents the distance between start and end points of the process.

(21) Ali *yedi km* yürüdü. (NP = measure)
Ali seven km walk+PAST+3SG
'Ali walked seven km-s.'

- **Origin** is a function that represents a location from where the process starts.

(22) Ali *okuldan* geldi. (NP+ablative)
Ali school+ABL come+PAST+3SG
'Ali came from school.'

- **Location** is a function that represents a location where the process is occurs .

(23) Ali *okulda* ders çalışıyor. (NP+locative)
Ali school+LOC study+PROG+3SG
'Ali is studying at school.'

- **Destination** is a function that represents a location where the process is directed towards.

(24) Ali *okula* gitti. (NP+dative)
Ali school+DAT go+PAST+3SG
'Ali went to school.'

- **Path** is a function that represents the track or way of the process.

(25) Ali okula *patika yoldan* gitti. (NP+ablative)
Ali school+DAT foot-path+ABL go+PAST+3SG
'Ali went to school via foot-path.'

2. *Extent-&-Location in time* (Temporal Functions)

- **Duration** is a function that represents the time during which the process continues.

  (26) a. *Üç gün* (*boyunca*) ders çalıştım. (NP or PP)
  three days during study+PAST+1SG
  'I have studied for three days.'

  b. Ali *1980'den beri* hastanede çalışıyor. (PP)
  Ali 1980+ABL since hospital work+PROG+3SG
  'Ali has been working at hospital since 1980.'

  c. Ali *sürekli* hikaye kitabı okur. (AdvG)
  Ali always story book read+AOR+3SG
  'Ali always reads story books.'

- **Frequency** is a function that represents the repeated or frequent happening of the process.

  (27) a. Ali *sık sık* kütüphaneye gider. (AdvG)
  Ali often library+DAT go+AOR+3SG
  'Ali often goes to library.'

  b. Ali *beş kere* telefon etti. (PP)
  Ali five time phone+PAST+3SG
  'Ali phoned five times.'

- **Time** is a function that represents the time of the process.

  (28) a. Ali okula *dün* gitti. (AdvG)
  Ali school+DAT yesterday go+PAST+3SG
  'Ali went to school yesterday.'

  b. Ben *1971'de* doğdum . (NP+locative)
  I 1971+LOC born+PAST+1SG
  'I was born in 1971.'

  c. *Saat 5'den önce* iki elma yedim. (PP)
  5 o'clock+ABL before two apple eat+PAST+1SG
  'I ate two apples before 5 o'clock.'

3. *Manner* indicates the way or style of doing something. They are called Process Adjuncts in [38]. They may be specified as follows.

   - **Means** represents a method or instrument that is used to do something.

(29) Ali mektubu *uçakla* gönderdi. (NP+instrumental)
Ali letter+ACC plane+INS send+PAST+1SG
'Ali sent the letter by air mail.'

- **Quality** explains how the process is performed. It is generally realized by adverb groups.

(30) Ali mektubu *dikkatsizce* yazdı. (AdvG)
Ali letter+ACC carelessly write+PAST+1SG
'Ali wrote the letter carelessly.'

- **Comparison** explains what the process looks like.

(31) Su *buz gibi* soğuktu. (PP)
water ice like cold+COP+PAST+3SG
'Water was cold like ice.'

4. *Cause* explains what caused to do something. It can be decomposed into the following three sub-categories.

- **Reason** represents why the process is performed.

(32) *Soğuk havadan dolayı*
maç ertelendi. (PP)
cold weather+ABL because of
match postpone+PASS+PAST+3SG
'The match was postponed because of the cold weather.'

- **Purpose** represents for what reason the process is performed.

(33) *Ders çalışmak için* kütüphaneye gittim. (PP)
study+CONV=NOUN+NOM for library+DAT go+PAST+1SG
'I went to library to study.'

- **Behalf** represents for whom the process is performed.

(34) *Oğlu için* bir araba aldı. (PP)
son+3SP for a car buy+PAST+3SG
'He bought a car for his son.'

5. *Accompaniment* is someone/something that represents the meaning 'and', 'or', 'not' as circumstantial. It is a form of joint participants in the process.

- **Comitative** "represents the process as a single instance of a process, although one in which two entities are involved" [14].

(35) +. Ahmet *Ali ile (birlikte)* geldi. (PP)
Ahmet Ali with come+PAST+3SG
'Ahmet came with Ali.'

-. Ali okula *şemsiyesiz* gitti. (NP+privative)
Ali school+DAT umbrella+PRIV go+PAST+3SG
'Ali went to school without an umbrella.'

- **Additive** "represents the process as two instance; here both entities clearly share the same participant function, but one of them presented circumstantially for the purpose of contrast" [14].

(36) +. *Ali gibi* Veli'de geldi. (PP)
Ali as well as Veli+LOC come+PAST+3SG
'Veli came as well as Ali.'

-. *Ali'nin yerine* Veli geldi. (PP)
Ali+GEN instead of Veli come+PAST+3SG
'Veli came instead of Ali.'

6. *Matter* explains the subject or topic about which the process is performed.

(37) *Sınav hakkında* ne dedi? (PP)
examination about what say+PAST+3SG
'what did he say about the examination?'

7. *Role* represents the meaning of 'be' (attribute or identity) in the form of a circumstance.

(38) Oraya *bir öğrenci olarak* gideceğim. (PP/AdvG)
there a student as go+FUT+3SG
'I will go there as a student.'

### 3.3.2 Interpersonal Representation

Interpersonal metafunction expresses the relationship between the speaker and the listener (presented in Section 3.2.1). Interpersonal metafunction can be specified as follows:

- tense (primary time–*time* & secondary time–*mode*)

- polarity (positive or negative)
- mood (declarative, interrogative etc.)
- description of the process (POT, APP, DUR or HASTE)

In Turkish, these functions are realized in the verbal group. For that reason, more information about them is presented in Section 3.4. The values of these functions may be extracted from very abstract information. However, we assume that text planner produces their grammatical values in our current implementation.

### 3.3.3 Textual Representation

Textual metafunction presents the ideational and interpersonal information as text in context. In the syntactic level, "S O V" may be given as a default order. However, in the semantic level, the ideational and interpersonal functions are ordered according to their conflation with the following specified textual functions:

$$\underbrace{Topic\ Neutral}_{Theme} \cdots \underbrace{Focus\ Process\ Background}_{Rheme}$$

These functions and some grammatical constraints have already been presented in Section 3.2.4.

## 3.4 Verbal Group

A verbal group is constructed on a lexical element called **base** that can be a *verb* or a *nominal group*.[4] In (39.a) and (39.b), those kinds of verbal groups are respectively exemplified.

(39) a. *Gitmeyeceğim.*
go+NEG+FUT+1SG

[4]Adjectives and nouns are nominal groups in Turkish.

Base+suffixes
'I will not go.'

b. *Güzel* *değildir.*
beautiful NegNoun+AOR+COP+3SG
Base NegNoun+suffixes
'It is not beautiful.'

The **base** is the single lexical element that is given for the formation of a verbal group. The other lexical elements such as `değil`, `mi`, `ol` and the relevant suffixes. the components of the verbal group, are determined and organized by the systemic-functional grammar to express appropriate meanings. So, this section presents the possible structures of the verbal groups and their internal organization in Turkish [2, 23].

There are more than one grammatical structure of the verbal groups to express many distinct meanings. Fortunately, they may be generalized according to the type of **base** (*nominal group, verb*) and the **mood** (*finite, non-finite*). The selected features from these two systems (*type-of-base* and *mood*) determine the appropriate structure for the verbal group. The selected features from the other systems in Figure 3.7 given in Section 3.4.2 organize the internal structure of the verbal group. As a result, the following general structures can occur:[5]

- if **base** is a *verb* and **mood** is *finite*
  This case is selected to realize the *process* of a verbal sentence, or question. The type of the process can be material or mental. The structure of verbal groups for this case is shown in Table 3.13.[6] There exists two distinct components of the verbal group for interrogative sentences (questions): **base** and **interrogative tag**. The *Mode, Person,* and *Number* are added to **base** or **interrogative tag** depending on the selected values of these functions.

[5]The structures are considered in the tabular forms. The center row of the table describes the required functional elements of the verbal group in a grammatical order. The top rows of the table give examples, and bottom rows present their grammatical values, respectively. All possible values of each element are presented in Table 4.4 given in Chapter 4.

[6]VF stands for Voice Frame; Pol for Polarity; DV for Descriptive Verb; DP for Descriptive Polarity M-P-N for Mode, Person, and Number; Pot for Potential; Pos for Positive; Neg for Negative.

| sev yaz | -dır | | -ebil | | -melisin -acak |
|---|---|---|---|---|---|
| Base | VF | Pol | DV | DP | Finite |
| Verb Verb | ... ... | Pos Pos | Pot none | Pos Pos | ... ... |

| mı | -y-dı |
|---|---|
| Interr-Tag | M-P-N |
| none yes-no | ... |

| | -dır | |
|---|---|---|
| Subj-Obj-rel | Transition | Voice |
| none none | none Trans 1 | Active Active |

Voice Frame

| -meli -acak | -dı | -sin | |
|---|---|---|---|
| Time | Mode | Person | Number |
| Necess Future | none Past | Second Third | Sing Sing |

Finite

Table 3.13: The Structure of Finite Verbal Group from Verb

(40) a. Arkadaşlarını *sevebilmelisin.*
friend+3PL+2PP+ACC love+POT+NECES+2SG
'You *ought to be able to love* your friends.'

b. Ali mektubu *yazdıracak* *mıydı?*
Ali letter+3SG+ACC write+CAUS+FUT Ques+PAST+3SG
'*Was* Ali *going to have* the letter *written*?'

- if **base** is a *verb* and **mood** is *non-finite*
The structure of finite verbal group of a verbal sentence (given above) can be used in this case by replacing the **finite** with a **non-finite** element. A non-finite verbal group realizes the *process* of a clause that may be used as a noun (infinitive), adjective (participle) or adverb (adverbial). As a result, the structure for this case is given in Table 3.14.

(41) a. Birisi tarafından *sevilmek* güzeldir.
someone by love+PASS+CONV=NOUN nice+COP+AOR+3SG
'*To be loved* by someone is nice.'

| | | | | | |
|---|---|---|---|---|---|
| Sev | -il | | | | -mek |
| Oku | | | | | -yacak |
| Koş | | | | | -arak |
| Base | Voice Frame | Polarity | Desc-Verb | DP | Non-Finite |
| Verb | ... | Pos | none | | mek (noun) |
| Verb | | Pos | none | | ecek (adjective) |
| Verb | ... | Pos | none | | arak (adverb) |

Table 3.14: The Structure of Non-Finite Verbal Group from Verb

b. Mektupu *okuyacak*
adam gelmedi.
letter+3SG+ACC read+CONV=ADJ
man+NOM come+NEG+PAST+3SG
'The man *who would read* the letter did not come.'

c. Ali okula *koşarak* gitti.
Ali school+DAT run+CONV=ADV go+PAST+3SG
'Ali went to school *by running.*'

- if **base** is a *nominal group* and **mood** is *finite*
This case is selected to realize the *relational processes* that express the way of "being." Here, the **base** is a nominal group that may be an attribute or identifier in a nominal sentence or question. The type of "being" may be intensive, circumstantial, or possessive. According to its type, the **base** may take some suffixes such as locative and possessive before the formation of the verbal group. In the generation of a verbal group, we assume that the **base** is a lexical element, and the required suffixes or the distinct elements are determined by the systemic grammar to express the appropriate meanings. This case involves two types of grammatical structures. One of them is selected to realize a relational process by depending on the value of the *Time*. In the first structure shown in Table 3.15, a *substantive (predicative)* verb like an auxiliary verb is attached to **base** to demonstrate the "to be" meaning of the *process*. In addition, a distinct element called **neg-noun** is located after **base** to express the negative meaning. In the second structure shown in Table 3.16, an auxiliary verb "`olmak`" appears as a separate element after the **base**. If the value of *Time* is *Aorist, Past, Narr,* or *Cond* then the first structure is selected, otherwise the second one is selected.

| | |
|---|---|
| Öğretmen | -dir |
| Öğretmen | – |
| Öğretmen | – |
| Base | Finite |
| Noun | substantive |
| Noun | – |
| Noun | – |

| | |
|---|---|
| – | – |
| değil | -dir |
| – | – |
| Pol | Finite |
| Pos | – |
| Neg | subst... |
| Pos | – |

| | |
|---|---|
| – | – |
| – | – |
| mi | -dir |
| Interr-Tag | Finite |
| none | – |
| none | – |
| yes-no | subst... |

Table 3.15: The Structure of Finite Verbal Group from Nominal Group (1)

(42) a. O bir *öğretmendir.*
He a teacher+COP+AOR+3SG
'He *is a teacher.*'

b. O bir *öğretmen değildir.*
He a teacher not+COP+AOR+3SG
'He *is not a teacher.*'

c. O bir *öğretmen midir?*
He a teacher Ques+COP+AOR+3SG
'*Is* he a *teacher?*'

| |
|---|
| Öğretmen |
| Base |
| Noun |

| |
|---|
| olmayacaktı |
| Aux:verbal-group, mood:finite |
| ... |

Table 3.16: The Structures of Finite Verbal Group from Nominal Group (2)

(43) a. Ali *öğretmen olmayacaktı.*
Ali teacher be+NEG+FUT+PAST+3SG
'Ali *was not going to be a teacher.*'

- if **base** is a *nominal group* and **mood** is *non-finite*
In this case, the same structure in Table 3.16 is used by changing the value of the **mood** of auxiliary verb with *non-finite.*

(44) a. yazar olmak
writer be+CONV=NOUN
'to be a writer'

b. yazar olan
writer be+CONV=ADJ
'(someone) who is a writer'

c. yazar olarak
writer be+CONV=ADV
'as a writer'

### 3.4.1 The Elements in the Verbal Group

**Base** is the main element of the verbal group. It may be a *verb* or a *nominal group*.

**Voice Frame** determines the different states of Subject and Direct Object in the sentence.

- **According to Subject-Object-Relation** (Subj-Obj-rel)
  If the effect of the process is on the person or thing that does it, subj-obj-rel is called *reflexive* (subject and object are same). If two participants do the same thing to each other, subj-obj-rel is called *reciprocal* (subject and object are mutual).

- **According to Direct Object** (Transition)
  If the process involves another person or thing (direct object) that the process affects, this process is called *transitive* otherwise it is called *intransitive*. Here we defined a function that named **lex-transition** to represent the transitive or intransitive feature of the given lexical verb. This value is obtained from the lexicon as an input for the verbal group.

  An intransitive verb can be made transitive by using one of causative suffixes (`-t, -Ir, -tUr, -dIr, -er, -ert`) in Turkish. In addition, a transitive verb can be made transitive again and again in higher degrees by using the same suffixes. These transformations are required to explain the causative relations[7] between the participants of a clause. The order of causative suffixes and their functions [39] are presented in Table 3.17.

  The systemic-grammar determines the required transformations at the clause level and uses them in the realization of the Process. We

[7] Subject causes something to be done by someone else.

| *verb* | *causative functions* | | |
|---|---|---|---|
| | transitive | agentive | intensive |
| $V_{intransitive}$ | -dir∼-t<br>-it<br>-ir<br>-er∼-ert | -dIr∼-t | -dIr∼-t |
| $V_{transitive}$ | | | |

Table 3.17: The Order of the Causative Suffixes and their Functions

have defined a **transition** function to represent these transformations in the verbal group.

Theoretically, there is no restriction on the transition degree of a verb. However, the listener and speaker may lose the the information in the clause after three transitions. So, the clauses that involve max three transitions are considered in this study.

- **According to Subject** (Voice)

  Many actions involve two participants–one that performs the action and the other one is affected by the action. These kinds of actions are called transitive above. Any one of these participants can be the subject in the clause. If we want to focus on the participant that is the performer of the action, we make it the subject. and we use the *active* form of the verb (45.b). Otherwise, the affected participant is made the subject of the verb, and its *passive* form (45.b) is used.

  (45) a. Ali camı *kırdı.* (active)
  Ali window+ACC break+PAST+3SG
  'Ali broke the window.'

  b. Cam Ali tarafından *kırıldı.* (passive)
  window Ali by break+PASS+PAST+3SG
  'The window was broken by Ali.'

**Polarity,** and **Descriptive Polarity**
These two polarities explain the positive and negative senses of the main and descriptive verb, respectively.

**Descriptive Verb** is a verb that is added to the stem of the main verb to explain the descriptive meanings of the process. The main verb and Descriptive verb may be positive or negative sense but only one of them can be negative at the same time. There are four types of descriptive verbs:

- **Potential Verb** (-bilmek, -Amam) explains whether something is possible or not.

  (46) a. Mehmet *gelebilirdi.*
  Mehmet come+POT+PAST+3SG
  'Mehmet was able to come.'

  b. Yarın okula *gelemeyeceğim.* (not eq. gelmeyeceğim)
  tomorrow school+DAT come+POS+POTneg+FUT+1SG
  'Tomorrow I will not be able to come to school.'

- **Verb of Haste** (-A(y), -I(y) + -ver)

  (47) Mektubu *yazıverdim.*
  letter+ACC write+HASTE+PAST+1SG
  'I immediately wrote the letter.'

- **Durative Aspect** (-A(y) + -durmak, -kalmak, -gelmek etc.)

  (48) Sen masayı *temizleyedur.*
  you table+ACC clean+DUR+IMP+2SG
  'Keep on cleaning the table.'

- **Approximative Verb** (-A + -yaz)

  (49) *düşeyazdım.*
  fall+APP+PAST+1SG
  'I was about to fall down.'

**Finite** is an element that includes the following functions into verbal group.

- **Time** Primary time
- **Mode** Secondary time
- **Person** Third Second First
- **Number** Singular Plural
- **Interr-Tag** In the polar questions, the interrogative tag (*mI/mU*) becomes a distinct part of the finite verbal group.

  (50) a. Onlar *geliyorlar* *mı?*
  they come+PROG+3PL Ques
  'Are they coming?'

b. Siz *geliyor* *musunuz?*
you come+PROG Ques+2PL
'Are you coming?'

The positions of the elements: Mode, Person, and Number are changed according to their values. For example, if Person is third (50.a), the Mode, Person, and Number are added to main verb. However, if Time is progressive and Person is second (50.b), person and number are added to Interr-Tag.

**Auxiliary Verb** is a verb that is combined with a noun or a noun group to make a process from that noun. The combination of noun and auxiliary verb is used as Base in the verbal group. In Turkish, the most used auxiliary verbs may be ordered as follows. But, we only consider the auxiliary verb "`olmak`" (to be) in the implementation.

- `Olmak` is used to explain a state or relation.

  (51) a. Mehmet *öğretmen olmalıydı.*
  Mehmet teacher be+NEC+PAST+3SG
  'Mehmet should have been a teacher.'

- `Etmek/Eylemek/Kılmak` are used to explain different actions.

  (52) Bu evi biz *inşa ettik.*
  this house+3SP+ACC we build+PAST+3PL
  'We built this house.'

**Negative-Noun** (Turkish word `değil`) is used to represent the negative meaning of the nominal verbal group.

**Non-Finite** is used to generate a non-finite verbal group that realizes the process of a non-finite clause. This element is a particular suffix that provides the conversation from verb to noun, adjective, or adverb. The non-finite element is determined by the type of the non-finite clause. There are three types of non-finite clauses: *infinitive, participle, adverbial.* The possible non-finite suffixes can be listed as shown in Table 3.18.

### 3.4.2 System Network of Verbal Group

In Figure 3.6, if we select the *clause* feature from the *rank* system, SFG introduces the *process* as a function of the clause, and then realizes it as a verbal

| *Type* | *Conversion to* | *Suffixes* |
|---|---|---|
| Infinitive | noun | -mek, -me, -iş |
| Participle | adjective | -dik, -miş, -diği (past) |
| | | -en, -()r, -mez (present) |
| | | -ecek, -ceği (future) |
| Adverbial | adverb | -ip, -ken, -erek, -ince ... |

Table 3.18: Non-finite Suffixes

group by re-entering the network. The selection of a feature from each sys-

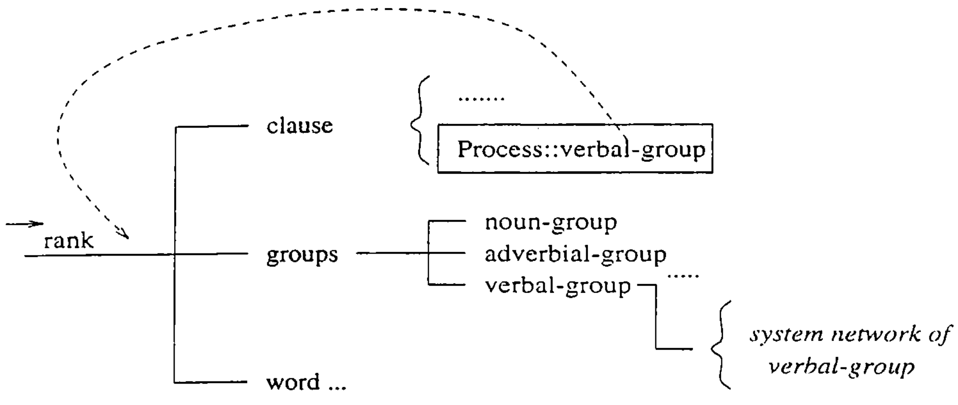

Figure 3.6: The Place of Verbal Group in SFG

tem, and the representation of realization rules depend on the implementation formalism. So, these issues will be considered in Chapter 4.

The required systems, the realization rules, and the appropriate context of each system in the linguistic description of the verbal group are determined and organized by using the analysis in the previous section. As a result, the system network given in Figure 3.7 is constructed.

In the network, only the systems and their appropriate contexts are displayed to express the basic linguistic description of the verbal groups. Because of this simplification, more specific rules and relations are not displayed in the network. However, they are considered and handled in the implementation.

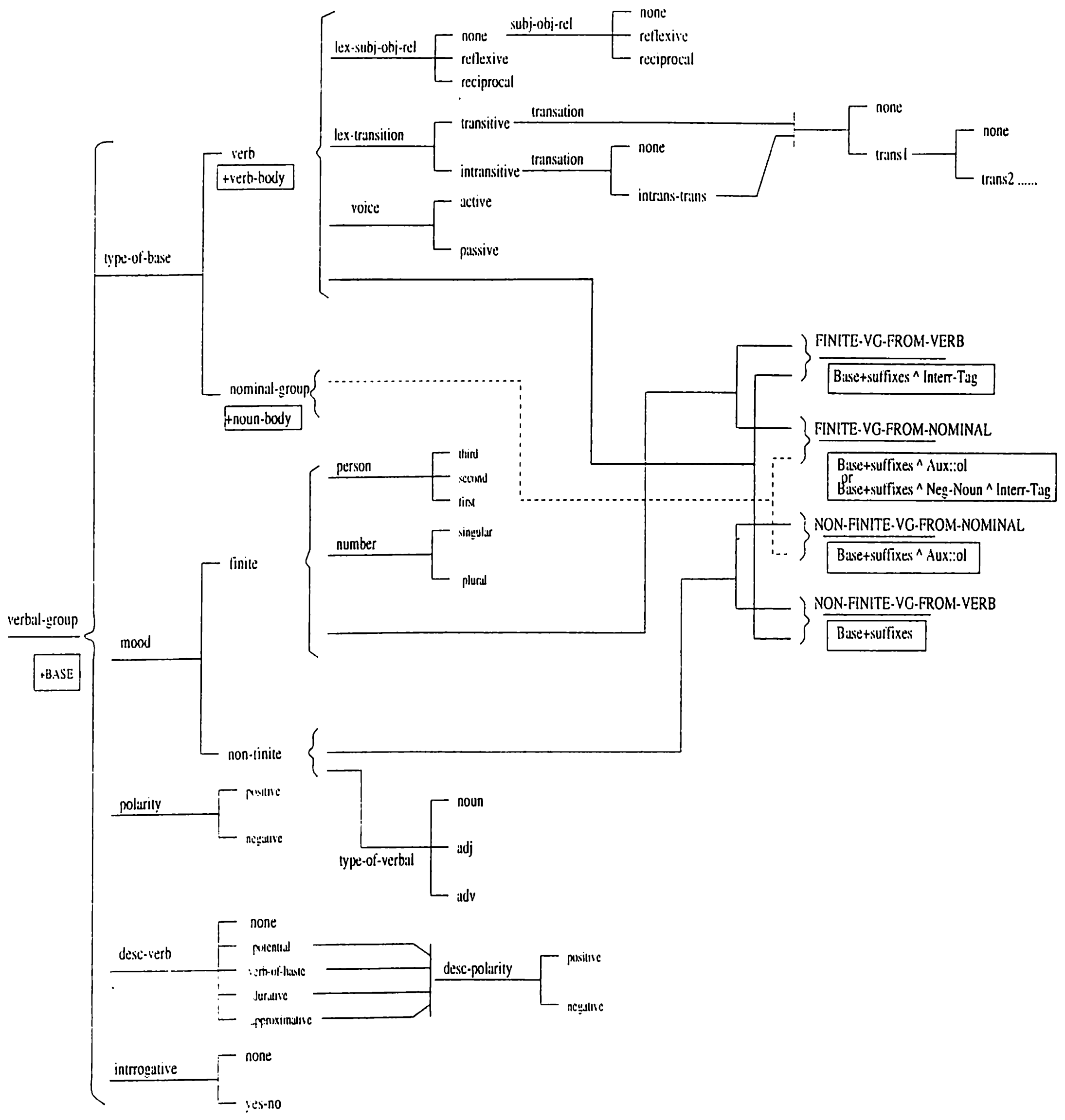

Figure 3.7: A System Network for Verbal Groups in Turkish

## 3.5 Noun Group (NP)

Noun Group (noun phrase–NP) is a grammatical unit that contains at least a noun called HEAD, and its modifiers.[8] The modifiers express the various types of the information about the head noun, but do not change its semantic features. Indeed, the semantic features of the head noun also belong to the entire NP [11]. Thus, the NP can be interpreted as an expansion of the head noun.

[8]Here, modifier represent all kind of elements that determine, describe, modify, or classify the head noun.

The head noun can be a common noun, a proper noun, or a pronoun. According to this choice, the head noun is modified by different grammatical functions that may be interpreted as the constituents of the NP. The general grammatical functions that expand the head noun can be described as follows [14]:[9]

**Head** *(Thing)*

**Determiner** *(Deictic & Numerative)* indicates whether a subset of the head noun is specific or not, and expresses the numerical features of the head.

**Describer** *(Epithet)* indicates the subjective and objective properties of the head noun.

**Classifier** *(Classifier)* indicates a particular subclass of the head noun.

**Qualifiers** *(Qualifiers)* indicate the characteristics of the head noun like the previous elements of the NP, but *"the characterization here is in terms of some process within which the thing (head noun) is, directly or indirectly, a participant"* [14]. They may be realized by a participle clause, or an oblique noun group.

These grammatical functions can be divided into more specific sub–functions. In the following sections, the specific functions and their partial orders will be considered and exemplified. Most of these functions can not be used if the head noun is a pronoun or a proper noun. However, all of them can be used with common nouns. Thus, to consider all functions, we assume that the head noun is common; and the pronoun and the proper noun phrases only consists of unmodified pronoun and proper noun, respectively.

[9]The grammar books use different names for these functions. In this document, we use the names in the traditional and the functional grammars together. The italic names are used in the functional grammar by Halliday.

### 3.5.1 Determiner

Determiner is decomposed into following elements:[10]

$$\underbrace{\overbrace{Possessor\ Demonstrative}^{Specific}\ \overbrace{nsDeictic}^{non-specific}}_{Deictic}\cdots\underbrace{Ordinal\ Quantitative}_{Numerative}\cdots\underbrace{Head}_{Thing}\underbrace{PostDet}$$

**Deictic** elements can be classified into two groups: specific or non-specific. They are presented in Table 3.19 and 3.20, respectively.

| | Determinative | Interrogative |
|---|---|---|
| Demonstrative | bu şu o | hangi |
| Possessive (Possessor) | benim senin onun<br>bizim sizin onların<br>Ali'nin | kimin<br>hangi (adam)ın |

Table 3.19: Specific Deictic

| | | Singular | Plural | Mass |
|---|---|---|---|---|
| additional (addDet) | no | | | |
| | yes | başka, diğer | başka | |
| total (nsDet) | none | | | |
| | positive | her | bütün | |
| | negative | hiçbir | | |
| partial (nsDet) | no | | | |
| | yes | (herhangi) bir | bazı | biraz |

Table 3.20: Non-Specific Deictic

**Possessor** indicates the owner of the head noun, and is realized by a noun phrase that is marked with genitive case. In addition, the agreement of the possessor and the possessive of the head noun must be same, and it is enforced by the grammar. Possessor specifies the head noun in two mutually exclusive types: Determinative (53.a) or Interrogative (53.b).

[10]The elements are represented in partial order: Three dots represent that different functions may be located.

(53) a. *Ali'nin* kitapları
Ali+3SG+GEN book+3PL+3SP+NOM
'Ali's books'

b. *Kimin* kitapları
who+3SG+GEN book+3PL+3SP+NOM
'Whose books'

**Demonstrative** indicates a particular subset of the head noun according to the distance between the speaker and the specified *thing*, and is realized by a demonstrative adjective. If distance is *near*, *far*, or *too-far*, the demonstrative will be `bu`, `şu`, or `o`. In contrast to English, there is no plural form of demonstrative adjectives in Turkish (54.a and 54.b). Demonstrative also specifies the head noun in two mutually exclusive types: Determinative (54.a and 54.b) or Interrogative (54.c).

(54) a. *Şu* kitap
that book+3SG+NOM
'That book'

b. *Şu* kitaplar
that book+3PL+NOM
'Those books'

c. *Hangi* kitap
which book+3PL+NOM
'which book'

**ns-Deictic** indicates the sense of all, none, or some unspecified subset of head noun, and it is realized by non-specific determiners (given in Table 3.20). ns-Deictic can be decomposed into following elements:

$$\underbrace{addDet\ nsDet}_{ns-Deictic}$$

(55) a. *Başka* *bir* kitap
another a book+3SG+NOM
'another book'

b. *Diğer* *bütün* kitaplar
other all book+3PL+NOM
'other books'

**Numerative** is either quantity or order, either exact or inexact as shown in Table 3.21.

| | | | | |
|---|---|---|---|---|
| Ordinal | exact | | | birinci, ikinci |
| | inexact | | | önceki, sonraki |
| Quantitative num-Seq | exact | cardinal | | bir, iki |
| | | distributive | | ikişer, üçer |
| | | fraction | | 5 / 4 |
| | inexact | indefinite | (quantifier) | az, çok |
| | | fuzzy | min | en az (exact num) |
| | | | approx | yaklaşık (exact num ) |
| | | | max | en çok (exact num) |
| | | range | | üç beş, |
| Quantitative partitive | | measure | number | (value) adet, tane |
| | | | length | (value) (unit) |
| | | | area | (value) (unit) |
| | | | volume | (value) (unit) |
| | | | container | (container) dolusu |
| | | | weight | (value) (unit) |
| | | typical | | (value) parca, dilim |

Table 3.21: Numeratives

**Ordinal** indicates the place of the head noun in order, and is realized by ordinative adjectives. It expresses either an exact place (56.a) in order or an inexact place (56.b).

(56) a. *Birinci* masa
first table+3SG+NOM
'First table'

b. *Sonraki* masa
next table+3SG+NOM
'Next table'

**Quantitative** expresses the numerical and partitive features of the head noun. Quantitative is decomposed into following elements:

$$\underbrace{numSeq\ partitive}_{Quantitative}$$

**numSeq** indicates the exact (57.a) or inexact (57.b) numerical features, and is realized by numerative expressions (given in Table 3.21).

(57) a. *Beş* masa
five table+3SG+NOM
'Five tables'

b. *En fazla* üç masa
maximum three table+3SG+NOM
'Maximum three tables'

**partitive** indicates the expression of quantity by means of partitive (58) (given in Table 3.21).

(58) Beş *kilo* elma
five kilo apple+3SG+NOM
'Five kilos of apple'

**Post–Det** can be one of the elements in Table 3.22. The head noun is marked with genitive case.

| type of post–det | value | example |
|---|---|---|
| total | positive | 'nın hep+si |
| | negative | 'nın hiçbir+i |
| multiplier | | 'nın (value) kat+ı |
| fraction | | 'nın (numerator)+de (denumerator)+ü |
| partitive | | 'nın 5 kilosu<br>'nın ikisi / üçü ... |

Table 3.22: Post Determiners

(59) Elmanın beş *kilosu*
apple+3SG+GEN five kilo+3SG+3SP+NOM
'Five kilos of apple'

### 3.5.2 Describer

Describer is decomposed into following elements:

$$\underbrace{\overbrace{Attitude}^{Opinion} \cdots \overbrace{Quality}^{Fact}}_{Epithet} \cdots \underbrace{Head}_{Thing}$$

**Attitude** expresses the subjective property of the head noun, and is realized by an adjective. In order, it tends to precede numerative (60).

| | | | | |
|---|---|---|---|---|
| (60) | Şu | *sevimli* | iki | kız |
| | that | lovely | two | girl+3SG+NOM |
| | 'those lovely two girls' | | | |

**Quality** is also decomposed into following elements that express the objective properties of the head noun:

$$age\ size\ color \cdots madeOf\ Head$$

In general, the objective describers (**age, size, color**) are located between determiner and classifier (61).

| | | | | |
|---|---|---|---|---|
| (61) | Benim | *eski* | *siyah* | çantam |
| | my | old | black | bag+3SG+1SP+NOM |
| | 'My old black bag' | | | |

**madeOf** indicates what the head noun is made of, and is realized by a material noun. It precedes the head noun immediately, and it is used mutually exclusive with classifier.

| | | | |
|---|---|---|---|
| (62) | siyah | *tahta* | masa |
| | black | wood | table+3SG+NOM |
| | 'black wood table' | | |

Describers may accept the following degrees of comparison: Similarity (63.a), Comparative (63.b), Superlative (63.c).

| | | | | |
|---|---|---|---|---|
| (63) a. | *Taş* | *kadar* | *sert* | yürek |
| | rock | as | hard | heart+3SG+NOM |
| | 'Heart as hard as rock' | | | |

b. *Taşdan daha sert* yürek
rock+ABL more hard heart+3SG+NOM
'Heart harder than rock'

c. *En güzel* kız
most beautiful girl+3SG+NOM
'The most beautiful girl'

### 3.5.3 Classifier

Classifier is realized by a nominal group (64) that represents a particular sub-class.

(64) *yazılım* aracı
software tool+3SG+3SP+NOM
'software tool'

It precedes the head immediately:

*Classifier Head*

Sometimes the same word (an adjective) may be used either as a describer or as a classifier, with a difference in meaning: e.g. *fast train* may mean "train that goes fast" (*fast* = describer) or "train classified as express" (*fast* = classifier) [14]. There is no sharp line between these two grammatical functions, but there are significant differences [14]:

- Classifiers do not accept degrees of comparison or intensity,
- Classifiers tend to be organized in mutually exclusive and exhaustive sets,
- Describers can be used as the complement of the *be clause*, but classifiers cannot [40].

### 3.5.4 Qualifiers

Qualifiers are realized by prepositional phrases or clauses, and located after the head noun in English. However, the order of qualifiers in Turkish NPs is determined according to the function of the qualifier. If it specifies the head noun, it is located at the beginning of the NP. If it describes the head noun, it is located between the determiner and the describer, and so on. In addition, the qualifiers in Turkish NPs are realized by oblique noun phrases (65), or non-finite clauses (66).

Qualifiers can be decomposed into more specific functions that specify, describe or classify the head noun from different perspectives. They can be described, and relatively ordered as follows:

$$qualset \cdots qualdesc \cdots qualloc \cdots possession \cdots \underbrace{Head}_{Thing}$$

**qual-set** expresses the superset of the head noun. It is realized by NP which is plural and in ablative case.

| | | |
|---|---|---|
| (65) | *Ögrencilerden* | Ali |
| | student+3PL+ABL | Ali |
| | 'Ali among the students' | |

**qual-desc** expresses some properties of the head noun in a process. It is realized by a participle clause.

| | | | |
|---|---|---|---|
| (66) | *Okula* | *giden* | çocuk |
| | school+DAT | go+CONV=ADJ | child+3SG+NOM |
| | 'The child who goes to school' | | |

**qual-loc** expresses the spatial or temporal location of the head noun. It may be realized by NP (67.a-b) or AdvG (67.c) that takes the **+ki** relative suffix. In addition, the NP must be in locative case.

| | | | |
|---|---|---|---|
| (67) | a. | *Okuldaki* | çocuk (spatial) |
| | | school+LOC+REL=ADJ | child+3SG+NOM |
| | | 'The child (who is) at the school' | |
| | b. | *1921'deki* | savaş (temporal) |
| | | 1921+LOC+REL=ADJ | war+3SG+NOM |
| | | 'The war in 1921' | |
| | c. | *Dünkü* | film (temporal) |
| | | yesterday+REL=ADJ | film+3SG+NOM |
| | | 'The film yesterday' | |

**possession** expresses what belongs or what does not belong to the head noun. To explain the positive or negative possession, it is realized by NP which is in munitive (+lI) or privative (+sIZ) case, respectively.

(68) a. *Kırmızı şapkalı* kız
red hat+MUN girl+3SG+NOM
'The girl with red hat'

b. *Kırmızı şapkasız* kız
red hat+PRIV girl+3SG+NOM
'The girl without red hat'

### 3.5.5 System Network of Noun Group

Noun groups are used to realize several functions in the clauses. For the linguistic description of a noun group, the required systems, and their appropriate contexts are presented as a system network in Figure 3.8.

## 3.6 Post–Positional Group (PP)

Post-positional Group (PP) has a simple structure that consists of an NP or infinitive, and a postposition particle:

$$NP\ \underbrace{PostPos}_{particle}$$

Particles are closed class of words such as **göre** (according to), **doğru** (towards), **sonra** (after) etc. A particle cannot refer to any concept but it provides to construct the relationships between the NP and the other constituents. Each particle may enforce the NP to be in a particular case. According to this feature, particles are divided in to four classes that the NP is in nominal case (**için, kadar, gibi**), dative case (**göre, karşı, doğru**), ablative case (**sonra, önce, başka**), or the particle is case-marked (**ön, arka, tarafından**) [2]. This feature of the particle can be provided from the lexicon. Each of them can be exemplified as follows.

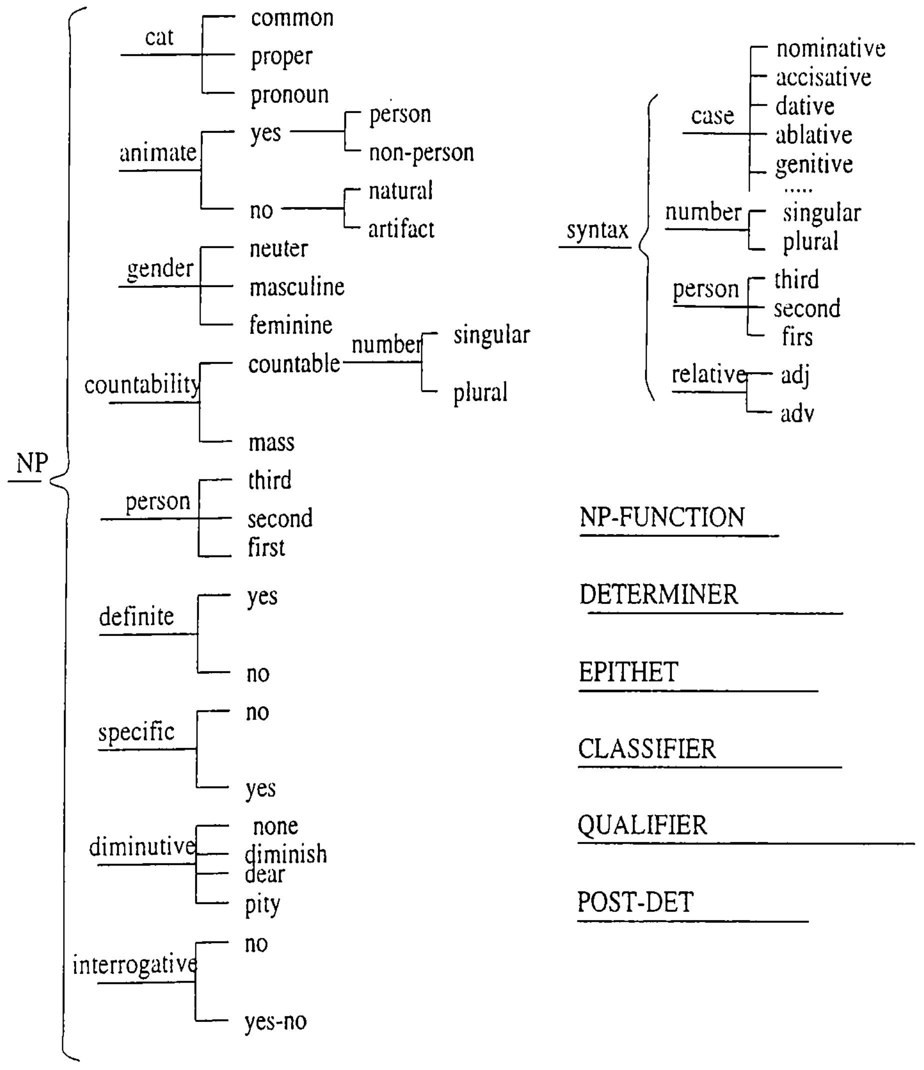

Figure 3.8: A System Network for Noun Group in Turkish

(69) a. *Kardeşi* *için* okula gitti.
brother+3SP for school+DAT go+PAST+3SG
'He went to school for his brother.'

b. *Ali'ye* *göre*, dünya yuvarlak değildir.
Ali+DAT according to world round NegNoun+COP+AOR+3SG
'According to Ali, the world is not round.'

c. *Sınavdan* *sonra* kütüphaneye gittim.
examination+ABL after library+3SG+DAT go+PAST+1SG
'I went to the library after the examination.'

d. *Arabanın önünde* iki çocuk vardı.
car+GEN front+3SP+LOC two child existent+COP+PAST+3SG
'There are two children in front of the car.'

PPs are used as adjuncts that realize the particular circumstantial functions in the clause (presented in Section 3.3).

## 3.7 Adverb Group (AdvG)

Adverb Group (AdvG) is used in the realization of several circumstantial functions that were illustrated in Section 3.3. The main constituent of an adverb group, Head, is an adverb that gives more information about when, how, where, or in what circumstances something happens or is done [14, 38]. In an adverb group, there may be additional modifiers to modify the head adverb. In the implementation, we have used a simple structure that contains only one modifier for the adverb group.

$$Modifier\ \underbrace{Head}_{adverb}$$

The modifier is also an adverb including the comparative daha (more), and the superlative en (most) forms.

(70) *Daha yavaş* konuşmalısın.
more slowly speak+NECES+2SG
'You must speak more slowly.'

According to the modification function, adverbs can be decomposed into several classes [2]:

- verification (evet, hayır, gerçekten)
- quantitive (az, çok)
- quality (duru, ince, uzun)
- place (aşağı, yukarı, dışarı)—place adverbs may take case suffix as shown in (71).

(71)

| | | |
|---|---|---|
| Seni | *dışarıda* | bekliyorum. |
| you+ACC | outside+LOC | wait+PROG+1SG |

'I am waiting for you outside.'

- temporal (`dün, sonra, önce`)
- frequency (`daima, bazen, sık sık`)
- manner (`hızlıca, sessizce`)

In the implementation, we have not considered the grammatical restrictions on adverb classes. We assume that the correct adverbs are given to realize the relevant circumstantial functions.

# Chapter 4

# Implementation

In order to develop a text generator with the Systemic–Functional Grammar, we need to implement the linguistic descriptions (system networks and realization rules) in a computational environment. For this purpose, we use the FUF text generation system, and its functional unification (FUG) and typed feature formalisms. In FUG framework, a data structure called functional description (FD) is handled. An FD is a list of pairs that each pair has an attribute name and value. Since we use the FUG formalism in the implementation [25], we need to translate the system network into this formalism. A system of the system network can be translated into disjunction of FDs, where each FD corresponds to an alternative in that system [20, 26]. Realization rules and relations between the systems are also translated into attribute–value pairs. This process is described by Kasper as an algorithm that translates SFG into FUG (for more information [20, 26]). In addition, FUF provides a typed feature formalism to implement the mutual exclusion, and hierarchical relations in SFG [3, 6, 21].

In this chapter, we present the basic architecture of the text generation system, and its implementation for Turkish. Then, we re-consider the linguistic descriptions and decisions in the generation of the major grammatical units such as clause, verbal group, noun group. These grammatical units have already been analyzed in the previous chapter.

## 4.1 Text Generation System

The FUF text generation system consists of two main modules: a **unifier** and a **linearizer** [7]. The unifier takes, as input, a *lexicalized semantic description* of the text to be generated, and an *extended form of FUG*, and then produces, as output, a *rich syntactic description* of the text or some new inputs[1] (the semantic and syntactic descriptions for the grammatical units that realize the specific components of the text) [8]. After the unification process, the linearizer takes the generated syntactic description as input, and then produces the morphological description of the text. At the end, a morphological generator may produce the worded text. As a result, the final text generation system can be organized as shown in Figure 4.1. In the given figure, the ellipses represent the

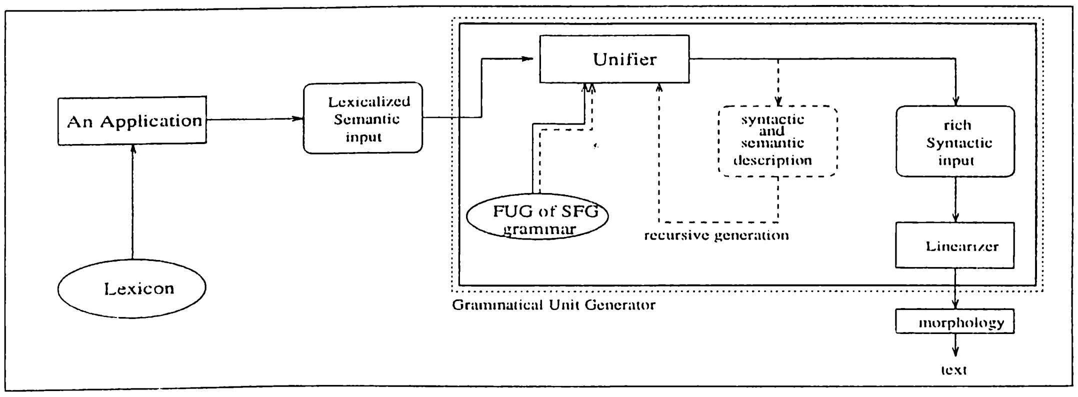

Figure 4.1: The Architecture of the Text Generator

linguistic resources; the boxes represent the computational operations; and the oval boxes represent the input/output descriptions.

In this thesis, we have tried to develop a Turkish text generator[2] by using the architecture given in Figure 4.1. In this generator, we have mainly considered the design and implementation of SFG for Turkish that will be presented in the next section. In addition, we have changed the morphology functions (e.g. `morph-verb, morph-noun`) of the linearizer for generating the morphological descriptions in Turkish. These morphological descriptions are the ultimate outputs of our current generator. They can be worded by the Morphological Analyzer/Generator presented in [36]. By the way, we assume

[1]These new inputs are produced and recursively performed by the unifier.

[2]In the Turkish text generator, the extra Turkish letters in the lexical items are represented as follows: C is ç, I is ı, G is ğ, O is ö, S is ş, U is ü.

that an application program that is not included in our implementation produces the *lexicalized semantic description* of the text.[3] Finally, the expected tasks from this generation system can be summarized as follows [7]:

- Determine the sentence configuration
- Map the semantic functions onto syntactic ones
- Provide ordering constraints
- Propagate agreement between the constituents
- Prevent over generation
- Select closed-class words
- Provide linguistic defaults etc.

## 4.2 Turkish Text Generation System with SFG

From the analysis given in the previous chapter, the main parts of Turkish SFG can be organized as shown in Figure 4.2. After transforming SFG into

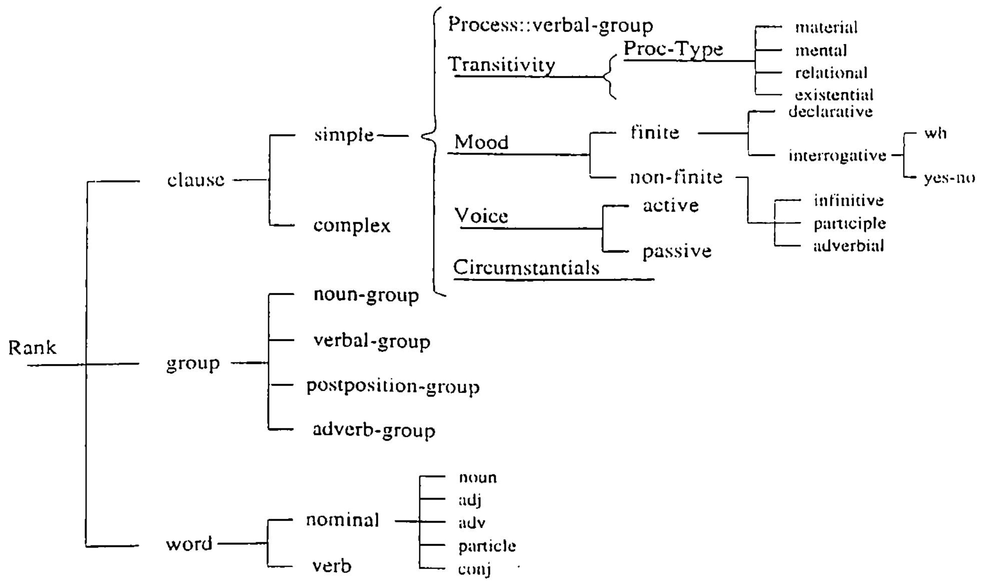

Figure 4.2: System Network of Turkish Grammar

[3]In the current generator, this input is given by hand.

FUG, the resulted grammar can be implemented in FUF as follows:[4]

```
;; gr-module.l
;;------------------------
(def-grammar gr ()
  (setq *realize-grammar*
   '((alt rank
      (((cat clause)
        (:! clause))
       ((cat group)
        (:! group))
       ((cat word)
        (:! word)))))))
;;------------------------
;; clause.l
;;------------------------
(def-alt clause (:index cat)
  (((:& simple-clause))
   ((:& complex-clause))))
(def-conj simple-clause
  (cat simple-clause)
  (participants ((...)))
  (:! mood)
  (:! transitivity)
  (:! voice)
  (process ((cat verbal-group) ...))
  (:! circumstantial)
  (pattern (... -textual-&-default-order- ....)))
;;-------------------------
;; group.l
;;-------------------------
(def-alt group (:index cat)
  (((:& verbal-group))                ; vg
   ((:& np))                          ; noun-group
   ((:& postposition-group))          ; pp
   ((:& adverb-group))))              ; advG
```

[4]The function **alt** is used to represent a system in FUF. The **def-alt** and **def-conj** functions provide us to write grammatical systems in a modular way. These modules are added to the main grammar by using the following operators :! and :&, respectively [8].

The whole grammar consists of the following modules:[5]

| | |
|---|---|
| `gr.l` | To load FUF System files and the grammar. |
| `types.l` | All type definitions used in the grammar. |
| `gr-modular.l` | Main module of implemented SFG for Turkish. |
| `clause.l` | Grammatical systems for clause. |
| `group.l` | Grammatical systems for group (phrase). |
| `word.l` | Grammatical words in Turkish. |
| `transitivity.l` | Transitivity system for Turkish. |
| `voice.l` | Voice system for Turkish. |
| `mood.l` | Mood System for Turkish. |
| `circumstantial.l` | Circumstantials mapped onto adjuncts. |
| `verbal-group.l` | Grammatical systems for verbal group. |
| `np.l` | Grammatical systems for NP. |
| `determiner.l` | Grammatical systems for determiners. |
| `linearize.l` | Linearizer produces morphological descriptions. |

To activate the text generation system for Turkish: FUF, and the Turkish SFG are loaded into Lisp.[6] In the Lisp environment, the semantic description of the text is given as a parameter to the uni function of FUF, next it produces a rich linguistic description, and then calls the linearizer to generate a morphological description for each constituent. Now, we will try to demonstrate the generation of a simple sentence in the implemented system. First, the semantic description of the sentence is given as follows:

```
;; ali gitti.
(uni '((cat simple-clause)
       (time past)
       (mood declarative) (voice active)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective no)
                 (lex "git")))
       (participants ((agent ((cat proper)
                              (lex "ali"))))))
```

[5]We have used the same modular approach in SURGE [7, 9].

[6]`gr.l` contains the required functions to load FUF and the grammar.

Next, in the unification step, FUF produces the following rich linguistic description that represents the mapping of the semantic function *agent* onto the syntactic one *subject*, the default word ordering constraint (Subject Verb), and the agreement between the subject and the verb:

```
((CAT SIMPLE-CLAUSE) (TIME PAST) (MOOD DECLARATIVE) (VOICE ACTIVE)
 (PROCESS
  ((TYPE MATERIAL) (TYPE-OF-BASE VERB) (AGENTIVE YES) (EFFECTIVE NO)
   (LEX "git") (voice active) (mood finite) (cat verbal-group)
   (polarity {polarity}) (desc-verb {desc-verb})
   (desc-polarity {desc-polarity}) (time {time}) (mode {mode})
   (person {synt-roles subject person})
   (number {synt-roles subject number})
   (verb-body
    ((cat verb) (lex {process lex}) (voice {process voice})
     (mood {process mood}) (polarity {process polarity})
     (time {process time}) (mode {process mode})
     (person {process person}) (number {process number})))
   (lex-transition transitive) (pattern (verb-body))))
 (PARTICIPANTS
  ((AGENT
    ((CAT PROPER) (LEX "ali") (np-function subject)
     (generic-cat group) (countability countable)
     (number singular) (person third) (np-type proper)
     (head
      ((cat noun) (definite yes) (lex {synt-roles subject lex})
       (person third) (number {synt-roles subject syntax number})
       (case {synt-roles subject syntax case})))
     (definite yes) (specific no) (referential no)
     (syntax ((case nominative) (person third)
              (number {synt-roles subject number})))
     (pattern  (dots {^ head} dots)) (case nominative)))
   (fset (actor agent range)) (actor {participants agent})))
 (generic-cat clause) (process-type {process type})
 (synt-roles
  ((fset (subject object iobject)) (subject {participants agent})))
 (pattern
  (st-topic (* {^ topic}) end-topic {^ synt-roles subject} dots st-focus
   (* {^ focus}) end-focus st-process {^ process} end-process dots
   st-background (* {^ background}) end-background dots))
 (polarity positive) (desc-polarity positive))
```

Then, in the linearizing step, FUF produces the morphological description of each constituent as follows:

```
[[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
[[CAT=VERB][ROOT=git][SENSE=POS][TAM1=PAST][AGR=3SG]].
```

These morphological descriptions can be worded by the Morphological Analyzer/Generator [36] as follows:

```
gener(ate)>  ali gitti.
```

## 4.3 Generation of Clauses

The semantic input of a clause consists of the given semantic functions in Table 4.1.[7] In the generation, the following types of linguistic decisions are required to produce a rich linguistic description of the clause.

- **Transitivity system** determines the acceptable participant configuration according to the process type. Transitivity and ergativity analysis (in Section 3.3.1) allow us to construct this system.
- **Voice system** determines the syntactic function of each participant which is determined by the transitivity system. There are two possible values of the voice: *active* and *passive*. We assume that the voice is given in the semantic input of the clause.
- **Mood system** determines the form of finite clauses such as statement, question etc., or the type of non-finite clauses such as noun, adjective, and

[7]Each function has already been described and exemplified in Section 3.3.

| Metafunctions | General Func. | More Specific Functions | Realization. |
|---|---|---|---|
| Ideational | Process | | VG |
| | Participants | agent, medium<br>actor, goal<br>beneficiary, range<br>senser, phenomenon<br>carrier, attribute<br>identified, identifier<br>entity (existential yes/no) | NP |
| | Circumstances | direction, distance, origin<br>location, destination, path<br>duration, frequency, time<br>means, quality, comparison<br>reason, purpose, behalf<br>comitative, additive<br>mater, role | NP,<br>PP,<br>AdvG |
| Interpersonal | mood | | in VG |
| Textual | voice | | in VG |
| | topic<br>focus<br>background | | conflated with<br>previous<br>functions |

Table 4.1: Semantic Description of a Clause

adverb. According to the *mood* function (finite or non-finite), additional functions may be added to the semantic description of the clause. They will be considered in the following sub-sections.

- **The order of constituents** is one of the most important aspects in the generation. The determination of the best order of the constituents may require more linguistic analysis. However, in our implementation, we use the exact order of the basic textual functions, and the partial default constituent lists to determine the relevant word order.[8] The order of textual functions have already been presented as follows:

  *Topic* ··· *Focus Process* ··· *Background*

[8]In FUF, the constituent order is given by a special function called `pattern`.

If a constituent is conflated with any one of these textual functions, it is located at the position of this textual function. Otherwise, the constituent is located according to a default order. In fact, it is not possible to describe an exact ordering list. However, we may describe the most used partial default ordering lists for the constituents such as

$$(pattern(subject \cdots object \cdots process))$$

$$(pattern(\cdots time \cdots location \cdots))$$

In FUF, these partial ordering lists are unified in a non-deterministic manner [8]. For instance, the given two lists can be unified as follows:

$$(pattern(subject \cdots time \cdots object \cdots location \cdots process))$$

$$(pattern(subject \cdots object \cdots time \cdots location \cdots process))$$

$$(pattern(subject \cdots time \cdots location \cdots object \cdots process))$$

In this way, the generator may produce the constituents in different orders by providing the partial default orders.

In the ordering list, participants are ordered according to their syntactic functions. However, circumstantials are directly ordered with their names. We have used the following partial default orders in the implementation:

$$subject \cdots by - object \cdots object \cdots iobject \cdots process$$

$$origin \cdots destination \cdots location \cdots direction \cdots distance \cdots path$$

$$time \cdots duration \cdots frequency \cdots process$$

$$means \cdots quality \cdots comparison \cdots process$$

$$reason \cdots purpose \cdots behalf \cdots process$$

$$additive \cdots comitative \cdots process$$

$$mater \cdots role \cdots process$$

So far, we have presented common features for the clauses. However, there is some distinction between finite and non–finite clauses. In the following two sub-sections, the other features will be considered.

### 4.3.1 Finite Clauses

In the finite clauses, *time* (primary time) and *mode* (secondary time) functions are included into the semantic description of the text. They are interpersonal functions, and also realized in the verbal-group. In addition, the finite clauses may be declarative, or interrogative. A declarative clause is generated as a statement (72) to give an information about something.

(72) Ali okula gitmişti.
Ali school+DAT go+NARR+PAST+3SG
'Ali had gone to school.'

The declarative clause given in (72) can be generated from the following semantic description:

```
input:
  (uni '((cat simple-clause)
         (time narr)          (mode past)
         (mood declarative) (voice active)
         (process ((type material) (type-of-base verb)
                   (agentive yes)  (effective no)
                   (lex "git")))
         (participants ((agent ((cat proper)
                                (lex "ali")))))
         (circum ((destination ((cat common)
                                (lex "okul")))))))
output:
  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DAT]]
  [[CAT=VERB][ROOT=git][SENSE=POS][TAM1=NARR][TAM2=PAST][AGR=3SG]].
```

An interrogative clause is generated as a question to get an information about something. In the interrogative clause, a *query-for* function appears in the semantic description, and must be conflated with the constituent that is inqueried. The type of the query may be *yes-no* (73.a) or *wh* (73.b).

(73) a. Ali mi okula gitti?
Ali Ques school+DAT go+PAST+3SG
'is it Ali who went to school?'

b. kim okula gitti?
who school+DAT go+PAST+3SG
'who went to school?'

These two clauses can be generated from the following semantic descriptions, respectively:

```
input for yes-no:
  (uni '((cat simple-clause)
         (time past)
         (mood yes-no) (voice active)
         (query-for {^ participants agent})
         (process ((type material) (type-of-base verb)
                   (agentive yes)  (effective no)
                   (lex "git")))
         (participants ((agent ((cat proper)
                                (lex "ali")))))
         (circum ((destination ((cat common)
                                (lex "okul")))))))
output:
  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
  [[CAT=QUES][ROOT=mI][AGR=3SG]]
  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DAT]]
  [[CAT=VERB][ROOT=git][SENSE=POS][TAM1=PAST][AGR=3SG]]?

input for wh:
  (uni '((cat simple-clause)
         (time past)
         (mood wh)  (voice active)
         (query-for {^ participants agent})
         (process ((type material) (type-of-base verb)
                  (agentive yes) (effective no)
                  (lex "git")))
          (circum ((destination ((cat common)
                                 (lex "okul"))))))
output:
  [[CAT=NOUN][ROOT=kim][AGR=3SG][POSS=NONE][CASE=NOM]]
  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DAT]]
  [[CAT=VERB][ROOT=git][SENSE=POS][TAM1=PAST][AGR=3SG]]?
```

Several examples are also presented in Appendix A.1.

### 4.3.2 Non-Finite Clauses

Non-finite clauses are constructed on a non-finite verbal group (considered in Section 4.4). According to the syntactic usage, non-finite clauses are divided into three types:[9] *infinitive, participle,* and *adverbial.* In a different grammatical unit, these clauses may be used as a noun, adjective, or adverb, respectively. In the generation, to provide the conversation from verb to noun, adjective, or adverb, we need to determine a *non-finite* element for the verbal group that realizes the process of non-finite clause. In fact, this information may be extracted from the semantic description by the grammar. However, our linguistic analysis is not enough to determine the non-finite elements for infinitives and adverbials in the grammar. So, we assume that the relevant non-finite elements (presented in Table 4.2) are given in the semantic description of these two type clauses.

| *mood* | *non-finite* |
|---|---|
| infinitive | mek, me, iş, meklik |
| adverbial | ip, arak, ken ... |

Table 4.2: Non-finite Elements for Infinitives and Adverbials

[9]In the semantic description, the *mood* function represents the type of non-finite clauses.

(74) Ali'nin okula gitmesi
Ali+GEN school+DAT go+CONV=NOUN=ME+3SG+3SP
'Ali's going to school'
'That Ali goes to school'

The infinitive clause given in (74) can be generated from the following semantic description:

```
input:
  (uni '((cat simple-clause)
         (mood infinitive) (non-finite me)
         (voice active)
         (process ((type material) (type-of-base verb)
                   (agentive yes)  (effective no)
                   (lex "git")))
         (participants ((agent ((cat proper)
                                (lex "ali")))))
         (circum ((destination ((cat common)
                                (lex "okul")))))))
output:
  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DAT]]
  [[CAT=VERB][ROOT=git][SENSE=POS][CONV=NOUN=ME]
   [TYPE=INFINITIVE][AGR=3SG][POSS=3SG][CASE=NOM]]
```

(75) okula giderek
school+DAT go+CONV=ADVERB=ARAK
'by going to school'

The adverbial clause given in (75) can be generated from the following semantic description:

```
input:
  (uni '((cat simple-clause)
          (mood adverbial) (non-finite arak)
          (voice active)
          (process ((type material) (type-of-base verb)
                    (agentive yes) (effective no)
                    (lex "git")))
          (circum ((destination ((cat common)
                                 (lex "okul"))))))
output:
  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DAT]]
  [[CAT=VERB][ROOT=git][SENSE=POS][CONV=ADVERB=ARAK]].
```

In the participle form, the grammar tries to determine the relevant non-finite element (presented in Table 4.3) according to the following criterion: the participle time, and the conflation of the modified or displaced constituent with subject.[10] In fact, there are several exceptions in the determination of the

| (mood participle) | | | |
|---|---|---|---|
| *participle* | *scope* conflated with | *non-finite* | example |
| future | subject | ecek | yazacak $adam_{Agent/Actor/Subject}$ yazılacak $mektup_{Medium/Subject}$ |
| | not subject | ecegi | yazacağım mektup |
| present | subject | en | yazan $adam_{Agent/Actor/Subject}$ yazılan $mektup_{Medium/Subject}$ |
| | not subject | — | — |
| past | subject | mis | yazmış $adam_{Agent/Actor/Subject}$ yazılmış $mektup_{Medium/Subject}$ |
| | not subject | digi | yazdığım mektup |

Table 4.3: Non-finite Elements for Participles

non-finite element in the participle if the *scope* is not conflated with *subject*. That kind of exceptions are considered in the tabular form for each displaced semantic constituent, and the resulted tables are presented in Appendix B. However, in the implementation we have not used them. They may be useful material for the future studies.

[10]The modified or displaced element is represented by a function called **scope**.

A participle clause is generated to describe one of its own constituents except for the process. In (76), the participle clause describes its own agent/actor.

(76) okula giden ....
school+DAT go+CONV=ADJ=EN ...
'... who goes to school'

The participle clause given in (76) can be generated from the following semantic description:

```
input:
  (uni '((cat simple-clause)
         (mood participle) (participle present)
         (scope {^ participants agent})
         (voice active)
         (process ((type material) (type-of-base verb)
                   (agentive yes)  (effective no)
                   (lex "git")))
         (participants ((agent ((cat proper)
                                (lex "ali")))))
         (circum ((destination ((cat common)
                                (lex "okul")))))))
output:
  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DAT]]
  [[CAT=VERB][ROOT=git][SENSE=POS][CONV=ADJ=EN]].
```

In the non-finite clauses, the textual function *background* is not used, so it is always set to none. In the infinitive clauses, the semantic role that is mapped onto subject[11] is realized in the genitive case (77).

[11]Technically, non-finite clauses have no subject. However, in the implementation we have not described another syntactic role to represent it.

| | | |
|---|---|---|
| (77) *Ali'nin* | kitabı | okumasını |
| ben | istedim. | |
| Ali+GEN | book+ACC | read+CONV=NOUN+3SP+ACC |
| I | want+PAST+1SG | |

'I wanted Ali's reading the book.'
'I wanted Ali read the book.'

Several non-finite clauses and their usage in the different grammatical units are presented in Appendix A.2.

## 4.4 Generation of Verbal Group

A verbal group realizes the following semantic functions of a clause: the *process* (action, event or relational process); the *interpersonal functions* (mood, tense, descriptive verb etc.); the *textual function* (voice). In the generation of a clause, the generator produces a linguistic description that contains the mentioned functions and additional syntactic ones for the verbal group. Then, the generator recursively uses it to generate the verbal group as a worded text. The required functions and their possible values in the linguistic description of the verbal group are presented in Table 4.4.

A set of these functions can be produced manually, and given as an input to the system.[12] Then, the system generates the verbal group from the given linguistic description by traversing the system network of the verbal group. In the input set, at least, the functions *cat*, *lex*, and *type-of-base* must be given with their values. The other functions are optional. If a function does not appear in the input set but it is required, the first alternative is selected as a default value for that function. The generation of the verbal groups can be exemplified as follows:

[12]In a real application, the verbal group is automatically generated in the realization of the clause.

| Condition | Function | Alternative values |
|---|---|---|
| | cat | verbal-group |
| | lex | *a lexical verb or nominal group* |
| | type-of-base | verb, nominal |
| if type-of-base is verb | lex-subj-obj-rel | none, reflexive, reciprocal |
| | lex-transition | transitive, intransitive |
| | subj-obj-rel | none, reciprocal, reflexive |
| | transition | none, intrans-trans trans1, trans2, trans3 |
| | voice | active, passive |
| | polarity | positive, negative |
| | desc-verb | potential, haste, durative, approximative |
| | desc-polarity | positive, negative |
| | mood | finite, non-finite |
| if mood is finite | time | aorist, past, narr, progress, future, cond, optative, necessitative, imperative |
| | mode | none, past, narr, cond |
| | person | first, second, third |
| | number | singular, plural |
| | interrogative | none, yes-no |
| if mood is non-finite | type-of-verbal | noun, adj, adv |
| | non-finite | mek, me, is (infinitive) |
| | | ecek, ecegi (participle future) |
| | | en (participle present) |
| | | mis, digi (participle past) |
| | | ip, ere, ince, dikce ... (adverbial) |

Table 4.4: The Input Functions for the Formation of Verbal Group

(78) *sevebilmelisin.*
love+POT+NEC+2SG
'You *ought to be able to love.*'

```
input:
  (uni '((cat verbal-group)    (lex "sev")
         (type-of-base verb)   (polarity positive)
         (desc-verb potential) (desc-polarity positive)
         (mood finite)         (time necessitative)
         (mode none)           (person second)
         (number singular)     (interrogative none)))
```

```
output:
 [[CAT=VERB][ROOT=sev][SENSE=POS][COMP=YABIL][TAM1=NECES][AGR=2SG]]
```

(79) *öğretmen olmayacaktı.*
teacher be+NEG+FUT+PAST+3SG
'S/he *was not going to be a teacher.*'

```
input:
  (uni '((cat verbal-group)      (lex "OGretmen")
         (type-of-base nominal)  (polarity negative)
         (desc-verb none)        (desc-polarity positive)
         (mood finite)           (time future)
         (mode past)             (person third)
         (number singular)       (interrogative none)))
output:
  [[CAT=NOUN][ROOT=OGretmen][AGR=3SG][POSS=NONE][CASE=NOM]]
  [[CAT=VERB][ROOT=ol][SENSE=NEG][TAM1=FUTURE][TAM2=PAST][AGR=3SG]]
```

If Time and Mode are PAST and NARR, respectively in the same verbal group, the generation will be failed because it is an ungrammatical combination.

```
input:
  (uni '((cat verbal-group)   (lex "kIr")
         (type-of-base verb)  (mood finite)
         (time past)          (mode narr)
         (person third)       (number singular)
         (interrogative none)))
<fail>
```

## 4.5 Generation of Noun Group (NP)

Noun groups realize several constituents of a clause such as agent, actor, location, time etc. The generator produces a linguistic description of the noun group from the semantic features and the grammatical relations of the realized constituent. Then, the generator recursively uses this linguistic description to generate a worded text. The grammatical analysis of the noun group was presented in Section 3.5. In this section, we will summarize this analysis and the syntactic features of the noun group in a tabular form to present the linguistic description of the noun group. We may consider the linguistic description of the noun group according to its semantic characteristics, the grammatical constituents, and the morphological features. The semantic characteristics of the entire noun group are inherited from its main element called ***head noun***. The lexicalized semantic description of the head noun is presented in Table 4.5.[13]

| Class | Semantic Func. | Alternative Values |
|---|---|---|
| Head | cat | common, pronoun, proper |
| | lex | *a lexical noun* |
| | animate | yes, no |
| | gender | neuter, masculine, feminine |
| | person | third. second, first |
| | countability | countable, mass |
| | number | singular, plural |
| if cat pronoun | pronoun-type | personal, demonstrative, relative, interrogative, indefinite |
| | definite | yes, no |
| | specific | no, yes |
| | diminutive | none, diminish, dear, pity |

Table 4.5: The Lexicalized Semantic Description of the Head Noun

The grammatical constituents of the noun group modify the head noun without changing its semantic features. These constituents and their syntactic realizations are presented in Table 4.6. More information about them is available in Section 3.5.

While the system network is being traversed to realize a semantic constituent with a noun group, the mentioned constituents and the lexicalized semantic features of the noun group are directly inherited from the input given

[13]The *cat*, *lex*, and *pronoun-type* functions represent the lexical features of the head noun.

| Class | Constituent | *Realization* and Sub-functions |
|---|---|---|
| Determiner | possessor | *cat np* |
| | demonstrative | *cat adj* |
| | | (type determinative) (distance near/far) |
| | | (type interrogative) |
| | ns-deictic | *(cat phrase)* **add-det ns-det** |
| | | (additional no/yes) |
| | | (total none/+/−) (partial yes/no) |
| | ordinal | *cat adj* (type ordinal) |
| | quantitative | *(cat phrase)* |
| | | (exact yes/no) (partitive no/yes) |
| | post-det | *(cat phrase)* (type total/multiplier/fraction) |
| Epithet | attitude | *cat adj* |
| | quality | consist of **age, size, color, made-of** realized by *adj, adj, adj, noun* |
| Classifier | classifier | *cat np, adj* (gradable no) |
| Qualifier | possession | *cat np* (it-is +/−) |
| | qual-possessor | *cat pp* (post-pos ((lex "ait")) |
| | qual-desc | *cat clause* (mood participle) |
| | qual-set-spec | *cat np* |
| | qual-loc | *cat np* (it-is spatial/temporal) |
| Inter-tag | inter-tag | (interrogative none/yes-no) |

Table 4.6: The Grammatical Constituents in the Noun Group

for the realized constituent. In addition, to provide the grammatical agreement, relations or conversation, some morphological features and the syntactic role of the noun group are determined, and added to the linguistic description by the grammar. All morphological features can be grouped under a syntactic function called *syntax*. The syntactic functions, and their possible values are presented in Table 4.7.

After producing the linguistic description of the noun group, the system network is re-entered to generate a morphological description for each constituent of the noun group by providing the ordering constraints and the grammatical rules. We may demonstrate the individual generation of the noun groups as follows:[14]

[14]Several examples are also presented in Appendix A.3.

| Class | Morphological Func. | Alternative Values |
|---|---|---|
| np-function | | subject, by-object, object, iobject, adjunct, predicate |
| Syntax | person | third, second, first |
| | number | singular, plural |
| | poss-person | none, third, second, first |
| | poss-number | none, singular, plural |
| | case | nominative, accusative, dative, locative, ablative, genitive, instrumental, equative, munitive, privative |
| | relative | adj, adv |
| | copula | none, yes |
| if | s-time | aorist, past, narr, cond |
| copula | s-person | third, second, first |
| is yes | s-number | singular, plural |

Table 4.7: The Syntactic Functions of the Noun Group

| (80) | Ali'nin | bazı | kitapları |
|---|---|---|---|
| | Ali+GEN | some | book+3PL+3SP |
| | ' some of Ali's books' | | |

```
input:
  (uni '((cat common)
         (lex "kitap")
         (definite no)
         (specific yes)
         (possessor ((number singular)
                     (person third)
                     (cat proper) (lex "ali")))
         (countability countable)
         (number plural)
         (ns-deictic ((partial yes)))))
output:
  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
  [[CAT=ADJ][ROOT=bazI]]
  [[CAT=NOUN][ROOT=kitap][AGR=3PL][POSS=3SG][CASE=NOM]].
```

| | | | |
|---|---|---|---|
| (81) | Benim | Bilkent'te bilgisayar | yazdığım programı |
| | I+GEN | Bilkent+LOC computer+3SG+NOM | write+CONV=ADJ=DIGI+1SP program+3SG+3SP+NOM |

'The computer program that I wrote at Bilkent'

```
input:
  (uni '((cat common)
         (lex "program")
         (definite yes)
         (classifier ((cat common)
                      (lex "bilgisayar")))
         (qualifier
           ((qual-desc
              ((cat simple-clause)
               (mood participle)
               (participle past)
               (process ((type material)
                         (lex-transition transitive)
                         (type-of-base verb)
                         (agentive yes)
                         (effective yes)
                         (lex "yaz")))
               (scope {^ participants medium})
               (participants ((actor ((cat pronoun)
                                      (person first)
                                      (number singular)
                                      (pronoun-type personal)))
                              (medium ((cat common)
                                       (lex "program")))))
               (circum ((location ((cat proper)
                                   (lex "bilkent"))))))))))
output:
;; [[CAT=NOUN][ROOT=ben][AGR=1SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=bilkent][AGR=3SG][POSS=NONE][CASE=LOC]]
;;  [[CAT=VERB][ROOT=yaz][SENSE=POS][CONV=ADJ=DIGI][POSS=1SG]]
;;  [[CAT=NOUN][ROOT=bilgisayar][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=program][AGR=3SG][POSS=3SG][CASE=NOM]].
```

Some functions may or may not appear in the linguistic description depending on other functions. For example, if *cat* is *proper*, then the *classifier* will be *none*; the *definite* and *specific* functions will be *yes*. Although we did not present those kinds of dependencies in the tables, they are handled in the implementation.

# Chapter 5

# Conclusion and Future Work

This thesis has presented the design and implementation of a Turkish text generation system based on SFG. In the design, we have tried to describe the linguistic resources of Turkish by using the following two major design principles of SFG [29]:

- Functional Analysis of NL
- The separation of paradigmatic (system-based) and syntagmatic (structure-based) organizations of NL.

In the functional analysis, generally we have used Halliday's work (presented in [14]) because all languages have common functions with a few differences. However, each language requires different paradigmatic and syntagmatic organizations to realize the functions of NL. So, we have analyzed Turkish grammar to provide its paradigmatic and syntagmatic organizations. As a result, we have constructed the system network of Turkish grammar, and described the realization rules that provide the relationships between the functional analysis and the syntactic structures, for each relevant feature in the system network. To represent and perform those systemic resources on computer, we have used the FUF text generation system and its constraint based formalisms such as FUG, and typed features [7]. So, we have translated the system network and the realization rules into the constraint based formalisms [7, 20], and implemented them in FUF [8].

The current text generation system takes as input the lexicalized semantic, and some system–based descriptions of the text at the sentence level, and then produces the morphological description of the text. The semantic description of a sentence consists of three metafunctions: *ideational* such as *agent, actor, goal, process, location*, for representing the constituents of the sentence and their roles, *interpersonal* such as *mood, modality*, for establishing the relationship between the speaker and the listener, and *textual* such as *topic, focus, background*, for presenting information as text in context. The system–based descriptions allow to enter the systems in the network, and to select the relevant feature from each system. The morphological description of the text, the output of the current generation system, is worded by the Turkish morphological generator [36].

For a complete text generation system, we need to develop a large-scale grammar based on Systemic–Functional theory. Unfortunately, there are no other Systemic–Functional studies on Turkish. Without such theoretical studies, it is not possible to develop a large grammar. Our current grammar allows us to generate simple sentences, noun phrases, verbal groups, post-positional groups and adverb groups. To generate more complex units, the functional analysis and systemic studies on Turkish must be considered at a larger scale. This may take a long time. For example, SURGE, the implementation of a large scale systemic functional grammar for English, is the result of five years of intensive experimentation in the grammar writing [7, 9]. However, we did not have such a long time in this thesis to develop a large-scale grammar of Turkish like SURGE. We have tried to show that "Turkish text can be generated by using systemic–functional approach even if it takes a long time." Fortunately, the systemic linguistic theory and the implementation tool (FUF) allow us to design and implement the grammar in a modular way. The whole grammar can be decomposed into small modules such as *clause, noun-group, verbal-group* as it is in the rank scale. Then, each module of the grammar is independently developed. In this way, the large scale grammar can be developed in a short time. In addition, to complete the overall Natural Language Generation system, the text planning stage must be considered while improving the grammar.

# Appendix A

# Sample Runs

In this chapter, we have presented sample runs to demonstrate what kind of grammatical units such as clause, noun-group etc. can be generated with our current text generator. Each sample is given as follows:

- Surface form of grammatical unit that will be generated.
- Its lexicalized semantic input.
- Its generated morphological description.

## A.1 Finite Clauses

### A.1.1 Declarative Sentences

```
;; ali uyuyabilmiSti.
(uni '((cat simple-clause)
       (time narr)        (mode past)
       (mood declarative) (voice active)
       (desc-verb potential)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "uyu")
```

```
                  (lex-transition intransitive)))
       (participants ((agent ((cat proper)
                              (lex "ali")))))))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=VERB][ROOT=uyu][SENSE=POS][COMP=YABIL]
;;  [TAM1=NARR][TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; turgay camI kIracaktI.
(uni '((cat simple-clause)
       (time future)     (mode past)
       (mood declarative) (voice active)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants ((actor  ((cat proper)
                               (lex "turgay")))
                      (medium ((cat common)
                               (definite yes)
                               (lex "cam")))))
       (topic {^ participants actor})
       (focus {^ participants medium})))
;; [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACCy]]
;; [[CAT=VERB][ROOT=kIr][SENSE=POS][TAM1=FUTURE]
;;  [TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; camI kIracaktI turgay.
(uni '((cat simple-clause)
       (time future)     (mode past)
       (mood declarative) (voice active)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants ((actor  ((cat proper)
                               (lex "turgay")))
                      (medium ((cat common)
                               (definite yes)
                              (lex "cam")))))
       (topic {^ participants medium})
       (focus none)
       (background {^ participants actor})))
;;  [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACCy]]
;;  [[CAT=VERB][ROOT=kIr][SENSE=POS][TAM1=FUTURE][TAM2=PAST][AGR=3SG]]
;;  [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]].
;;  =-=-=-=-=-=
;; cam turgay tarafIndan kIrIlacaktI.
(uni '((cat simple-clause)
       (time future)      (mode past)
       (mood declarative) (voice  passive)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants ((actor  ((cat proper)
                              (lex "turgay")))
                      (medium ((cat common)
                              (definite yes)
                              (lex "cam")))))
       (topic {^ participants medium})
       (focus {^ participants actor})))
;;  [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  tarafIndan
;;  [[CAT=VERB][ROOT=kIr][VOICE=PASS][SENSE=POS][TAM1=FUTURE]
;;   [TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; turgay  ali'ye  kIrdIrmalIydI camI.
(uni '((cat simple-clause)
       (time necessitative) (mode past)
       (mood declarative)   (voice  active)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants ((actor  ((cat proper)
                              (lex "ali")))
```

```
                       (agent  ((cat proper)
                                (lex "turgay")))
                       (medium ((cat common)
                                (definite yes)
                                (lex "cam")))))
       (topic {^ participants agent})
       (focus {^ participants actor})
       (background {^ participants medium})))
;; [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=DAT]]
;; [[CAT=VERB][ROOT=kIr][VOICE=CAUS][SENSE=POS][TAM1=NECES]
;;  [TAM2=PAST][AGR=3SG]]
;; [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACC]].
;;=-=-=-=-=-=
; turgay kitaplarI severdi.
(uni '((cat simple-clause)
       (time aorist)     (mode past)
       (mood declarative) (voice active)
       (process ((type mental)
                 (lex "sev")
                 (type-of-base verb)))
       (participants ((senser ((cat proper)
                                (lex "turgay")))
                      (phenomenon ((cat common)
                                   (number plural)
                                   (lex "kitab")))))))
;; [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=kitab][AGR=3PL][POSS=NONE][CASE=ACCy]]
;; [[CAT=VERB][ROOT=sev][SENSE=POS][TAM1=AORIST]
;;  [TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; turgay OGretmen olacaktI.
(uni '((cat simple-clause)
       (time future)             (mode past)
       (mood  declarative) (voice active)
       (process ((type intensive)
                 (rel-mode identifying)))
       (participants ((identified ((cat proper)
                                   (lex "turgay")))
                      (identifier ((cat common)
                                   (lex "OGretmen")))))))
;; [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=OGretmen][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=VERB][ROOT=ol][SENSE=POS][TAM1=FUTURE]
;;  [TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; bu kitab siyah deGilmiS.
(uni '((cat simple-clause)
       (time narr)
       (mood declarative) (voice active)
       (polarity negative)
       (process ((type intensive)
                 (rel-mode attributive)))
       (participants ((carrier   ((cat common)
                                   (lex "kitab")
                                   (demonstrative
                                     ((distance near)))))
                      (attribute ((cat adj)
                                  (lex "siyah")))))))
;; [[CAT=ADJ][ROOT=bu]]
;; [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=ADJ][ROOT=siyah]]
;; [[CAT=NOUN][ROOT=deGil][AGR=3SG][POSS=NONE][CASE=NOM]
;;  [CONV=VERB=NONE][TAM1=NARR][AGR=3SG]].
;;=-=-=-=-=-=-=
;; bu kitab siyahtI.
(uni '((cat simple-clause)
       (time  past)
       (mood declarative) (voice active)
       (polarity positive )
       (process ((type intensive)
                 (rel-mode attributive)))
       (participants ((carrier ((cat common)
                                (lex "kitab")
```

```
                              (demonstrative ((distance near)))))
                    (attribute ((cat adj)
                                (lex "siyah")))))))
;; [[CAT=ADJ][ROOT=bu]]
;; [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=ADJ][ROOT=siyah][CONV=VERB=NONE][TAM1=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; turgay okuldaydI.
(uni '((cat simple-clause)
       (time past)
       (mood declarative) (voice active)
       (process ((type spatial)
                 (rel-mode attributive)))
       (participants ((located ((cat proper) ; carrier
                                (lex "turgay")))
                      (location ((cat common)
                                 (lex "okul")))))))
;; [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=LOC]
;;  [CONV=VERB=NONE][TAM1=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; turgay okulda olacakmIS.
(uni '((cat simple-clause)
       (time future)      (mode narr)
       (mood declarative) (voice active)
       (process ((type spatial)
                 (rel-mode attributive)))
       (participants ((located ((cat proper) ; carrier
                                (lex "turgay")))
                      (location ((cat common)
                                 (lex "okul")))))))
;; [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=LOC]]
;; [[CAT=VERB][ROOT=ol][SENSE=POS][TAM1=FUTURE][TAM2=NARR][AGR=3SG]].
;;=-=-=-=-=-=
;; ali bu kitaba sahip olacaktI.
(uni '((cat simple-clause)
        (time future)      (mode past)
        (mood  declarative) (voice active)
        (process ((type possessive)
                  (rel-mode attributive)))
        (participants ((possessed ((cat common)
                                   (lex "kitab")
                                   (demonstrative
                                     ((distance near)))))
                       (possessor ((cat proper)
                                   (lex "ali")))))))
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=ADJ][ROOT=bu]]
;;  [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=NONE][CASE=DAT]]
;;  [[CAT=NOUN][ROOT=sahip][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=VERB][ROOT=ol][SENSE=POS][TAM1=FUTURE][TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; UC kitap vardI.
(uni '((cat simple-clause)
       (time past)
       (mood declarative) (voice active)
       (process ((type existential)
                 (existential yes)
                 (polarity positive)))
       (participants ((entity ((cat common)
                               (lex "kitap")
                               (quantitative
                                 ((exact yes)
                                  (value ((lex "UC")))
                                  (quan-type cardinal))))))))
;;  [[CAT=ADJ][ROOT=UC]]
;;  [[CAT=NOUN][ROOT=kitap][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=var][AGR=3SG][POSS=NONE][CASE=NOM]
;;   [CONV=VERB=NONE][TAM1=PAST][AGR=3SG]].

;;=======================================================
;; W i t h   C i r c u m s t a n t i a l s
;;=======================================================
;; ali okula gitmiSti.
(uni '((cat simple-clause)
       (time narr)        (mode past)
       (mood declarative) (voice active)
       (process ((type material) (type-of-base verb)
                 (agentive yes) (effective no)
                 (lex-transition intransitive)
                 (lex "git")))
       (participants ((agent ((cat proper)
                              (lex "ali")))))
       (circum ((destination ((cat common)
                              (lex "okul")))))))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DAT]]
;; [[CAT=VERB][ROOT=git][SENSE=POS][TAM1=NARR][TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;turgay daGa doGru yUrUyecekti
(uni '((cat simple-clause)
       (time future)       (mode past)
       (mood  declarative) (voice active)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective no)
                 (lex "yUrU")))
       (participants ((actor ((cat proper)
                              (lex "turgay")))))
       (circum ((direction ((cat  pp)
                            (np ((cat common)
                                 (lex "daG")))))))))
;; [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=daG][AGR=3SG][POSS=NONE][CASE=DAT]]
;; doGru
;; [[CAT=VERB][ROOT=yUrU][SENSE=POS][TAM1=FUTURE][TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
; turgay yedi km yUrUyecektI.
(uni '((cat simple-clause)
       (time future)       (mode past)
       (mood  declarative) (voice active)
```

```
         (process ((type material) (type-of-base verb)
                   (agentive yes)  (effective no)
                   (lex "yUrU")))
         (participants ((actor ((cat proper)
                                (lex "turgay")))))
         (circum ((distance ((cat phrase)
                             (exact yes)
                             (quan-type cardinal)
                             (value ((lex "yedi")))
                             (unit ((lex "km")))))))
         (focus  {^ circum distance})))
 ;;  [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
 ;;  [[CAT=ADJ][ROOT=yedi]]
 ;;  [[CAT=NOUN][ROOT=km][AGR=3SG][POSS=NONE][CASE=NOM]]
 ;;  [[CAT=VERB][ROOT=yUrU][SENSE=POS][TAM1=FUTURE][TAM2=PAST][AGR=3SG]].
 ;;=-=-=-=-=-=
 ;ankara'dan beri aGaC yoktu yolda.
 (uni '((cat simple-clause)
        (time past)
        (mood declarative) (voice active)
        (process ((type existential)
                  (existential no)))
        (participants ((entity ((cat common)
                                (lex "aGaC")))))
        (circum ((origin ((cat pp)
                          (post-pos ((lex "beri")))
                          (np ((cat proper)
                               (lex "ankara")))))
                 (location ((cat common)
                            (lex "yol")))))
        (topic {^ circum origin})
        (background {^ circum location})
        (focus {^ participants entity})))
;;  [[CAT=NOUN][ROOT=ankara][AGR=3SG][POSS=NONE][CASE=ABL]]
;; beri
;;  [[CAT=NOUN][ROOT=aGaC][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=yok][AGR=3SG][POSS=NONE][CASE=NOM]
;;  [CONV=VERB=NONE][TAM1=PAST][AGR=3SG]]
;; [[CAT=NOUN][ROOT=yol][AGR=3SG][POSS=NONE][CASE=LOC]].
;;=-=-=-=-=-=
; ali dUn kardeSi iCin gitti okula.
(uni '((cat simple-clause)
       (time past)
       (mood declarative) (voice active)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective no)
                 (lex "git")))
       (participants ((actor ((cat proper)
                              (lex "ali")))))
       (focus {^ circum behalf})
       (background {^ circum destination})
       (circum ((destination ((cat common)
                              (lex "okul")))
                (behalf ((cat pp)
                         (np ((cat common)
                              (lex "kardeS")
                              (possessor ((gap yes)
                                          (cat proper)
                                          (lex "ali")))))))
                (time ((cat adv)
                       (lex "dUn")))))
       (focus {^ circum behalf})
       (background {^ circum destination})))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=ADVERB][ROOT=dUn]]
;; [[CAT=NOUN][ROOT=kardeS][AGR=3SG][POSS=3SG][CASE=NOM]]
;; iCin
;; [[CAT=VERB][ROOT=git][SENSE=POS][TAM1=PAST][AGR=3SG]]
;; [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DAT]].
;;=-=-=-=-=-=
; ali'nin baSI sInavdan dolayI/OtUrU  aGrIdI.
(uni '((cat simple-clause)
       (time past)
       (mood declarative) (voice active)
       (process ((type material) (type-of-base verb)
                 (agentive no)   (effective yes)
                 (lex "aGrI")))
       (participants ((medium ((cat common)
                               (lex "baS")
                               (possessor ((cat proper)
                                           (lex "ali")))))))
       (circum ((time ((cat adv)
                       (lex "dUn")))
                (reason ((cat pp)
                         (np ((cat common)
                              (lex "sInav")))))))))
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;;  [[CAT=NOUN][ROOT=baS][AGR=3SG][POSS=3SG][CASE=NOM]]
;;  [[CAT=ADVERB][ROOT=dUn]]
;;  [[CAT=NOUN][ROOT=sInav][AGR=3SG][POSS=NONE][CASE=ABL]]
;;  OtUrU
;;  [[CAT=VERB][ROOT=aGrI][SENSE=POS][TAM1=PAST][AGR=3SG]].

;;==========================================================
;; C a u s a t i v e   r e l a t i o n s
;;==========================================================
;; ali CocuGu uyutabilmiSti.
(uni '((cat simple-clause)
       (time narr)        (mode past)
       (mood declarative) (voice active)
       (desc-verb potential)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "uyu")
                 (lex-transition intransitive)))
       (participants ((agent ((cat proper)
                              (lex "ali")))
                      (actor  ((cat common)
                               (lex "Cocuk")))))))
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=Cocuk][AGR=3SG][POSS=NONE][CASE=ACC]]
;;  [[CAT=VERB][ROOT=uyu][VOICE=CAUS][SENSE=POS][COMP=YABIL]
;;   [TAM1=NARR][TAM2=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; dUn  turgay  mektubu okulda dikkatlice  yazdIramayabilirdi.
;; dUn  turgay  okulda mektubu dikkatlice  yazdIramayabilirdi.
(uni '((cat simple-clause)
       (time aorist)        (mode past)
       (mood declarative)    (voice  active)
       (desc-verb potential) (desc-polarity negative)
       (voice  active)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "yaz")))
       (participants ((actor  ((cat proper)
                               (gap yes)
                               (lex "????")))
                      (agent  ((cat proper)
                               (lex "turgay")))
                      (medium ((cat common)
                               (definite yes)
                               (lex "mektup")))))
       (circum ((location ((cat common)
                           (lex "okul")))
                (time ((cat adv)
                       (lex "dUn")))
                (quality ((cat adv)
                          (lex "dikkatlice")))))
       (topic {^ circum time})
       (focus {^ circum quality})
       (background none)))
;;  [[CAT=ADVERB][ROOT=dUn]]
;;  [[CAT=NOUN][ROOT=turgay][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=LOC]]
;;  [[CAT=NOUN][ROOT=mektup][AGR=3SG][POSS=NONE][CASE=ACC]]
;;  [[CAT=ADVERB][ROOT=dikkatlice]]
;;  [[CAT=VERB][ROOT=yaz][VOICE=CAUS][SENSE=POS][SENSE=NEGC]
;;   [TAM1=AORIST][TAM2=PAST][AGR=3SG]].
```

### A.1.2 Interrogative Sentences

```
;;;============================================================
;;; mood -- INTERROGATIVE -- yes-no or wh
;;; (query-for  ...) indicates what it is in question.
;;;============================================================
;; ali mi okula gitti?
(uni '((cat simple-clause)
       (time past) (mood yes-no) (voice active)
       (query-for {^ participants agent})
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective no)
                 (lex "git")))
       (participants ((agent ((cat proper)
                              (lex "ali")))))
       (circum ((destination ((cat common)
                              (lex "okul")))))))
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=QUES][ROOT=mI][AGR=3SG]]
;;  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DATy]]
;;  [[CAT=VERB][ROOT=git][SENSE=POS][TAM1=PAST][AGR=3SG]].
;;=-=-=-=-=-=
;; ali okula gidecek miydi?
(uni '((cat simple-clause)
       (time future) (mode past)
       (mood yes-no) (voice active)
       (query-for {^ process})
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective no)
                 (lex "git")))
       (participants ((agent ((cat proper)
                              (lex "ali")))))
       (circum ((destination ((cat common)
                              (lex "okul")))))))
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DATy]]
;;  [[CAT=VERB][ROOT=git][SENSE=POS][TAM1=FUTURE][AGR=3SG]]
;; [[CAT=QUES][ROOT=mI][TAM2=PAST][AGR=3SG]]?
;;=-=-=-=-=-=
;; kim okula gitti?
(uni '((cat simple-clause)
       (time past)
       (mood wh)   (voice active)
       (query-for {^ participants agent})
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective no)
                 (lex "git")))
       (participants (;;! query for agent
                      (agent ((cat np)))))
       (circum ((destination ((cat common)
                              (lex "okul")))))))
;; [[CAT=NOUN][ROOT=kim][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=DATy]]
;; [[CAT=VERB][ROOT=git][SENSE=POS][TAM1=PAST][AGR=3SG]]?
;;=-=-=-=-=-=
;; camI dUn mU kIrmalIydI ensar?
(uni '((cat simple-clause)
       (time necessitative) (mode past)
       (mood yes-no)        (voice active)
       (query-for {^ circum time})
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants ((actor ((cat proper)
                              (lex "ensar")))
                      (medium  ((cat common)
                                (definite yes)
                                (lex "cam")))))
       (circum ((time ((cat common)
                       (lex "dUn")))))
       (background  {^ participants actor})
       (topic {^ participants goal})))
;; [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACC]]
;; [[CAT=NOUN][ROOT=dUn][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=QUES][ROOT=mI][AGR=3SG]]
;; [[CAT=VERB][ROOT=kIr][SENSE=POS][TAM1=NECES][TAM2=PAST][AGR=3SG]]
;; [[CAT=NOUN][ROOT=ensar][AGR=3SG][POSS=NONE][CASE=NOM]]?
;;=-=-=-=-=-=
;; okulun odunu ali tarafIndan mI kIrIlacakmIS?
(uni '((cat simple-clause)
       (time future) (mode narr)
       (mood yes-no) (voice  passive)
       (query-for {^ participants actor})
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants ((actor ((cat proper)
                              (lex "ali")))
                      (medium  ((cat common)
                                (definite yes)
                                (lex "odun")
                                (possessor ((cat common)
                                            (lex "okul")))))))
       (topic {^ participants medium})))
;; [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=odun][AGR=3SG][POSS=3SG][CASE=NOM]]
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;; tarafIndan
;; [[CAT=QUES][ROOT=mI][AGR=3SG]]
;; [[CAT=VERB][ROOT=kIr][VOICE=PASS][SENSE=POS][TAM1=FUTURE]
;;  [TAM2=NARR][AGR=3SG]]?
;;=-=-=-=-=-=
;; okulun odunu kim tarafIndan kIrIlacakmIS.
(uni '((cat simple-clause)
       (time future) (mode narr)
       (mood wh)     (voice  passive)
       (query-for {^ participants actor})
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants (;; ! query for actor
                      (actor  ((cat np)))
                      (medium ((cat common)
                               (definite yes)
                               (lex "odun")
                               (possessor ((cat common)
                                           (lex "okul")))))))
       (topic {^ participants medium})))
;; [[CAT=NOUN][ROOT=okul][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=odun][AGR=3SG][POSS=3SG][CASE=NOM]]
;; [[CAT=NOUN][ROOT=kim][AGR=3SG][POSS=NONE][CASE=NOM]]
;; tarafIndan
;; [[CAT=VERB][ROOT=kIr][VOICE=PASS][SENSE=POS][TAM1=FUTURE]
;;  [TAM2=NARR][AGR=3SG]]?
;;=-=-=-=-=-=
; bu kitaplarI mI sevecekmIS ali?
(uni '((cat simple-clause)
       (time future) (mode narr)
       (mood yes-no) (voice active)
       (query-for {^ participants phenomenon})
       (process ((type mental) (type-of-base verb)
                 (lex "sev")))
       (participants ((senser ((cat proper)
                               (lex "ali")))
                      (phenomenon
                         ((cat common)
                          (number plural)
                          (lex "kitab")
                          (demonstrative ((type determinative)
                                          (distance near)))))))
       (topic {^  participants phenomenon})
       (background {^ participants senser})))
;; [[CAT=ADJ][ROOT=bu]]
;; [[CAT=NOUN][ROOT=kitab][AGR=3PL][POSS=NONE][CASE=ACCy]]
;; [[CAT=QUES][ROOT=mI][AGR=3SG]]
;; [[CAT=VERB][ROOT=sev][SENSE=POS][TAM1=FUTURE][TAM2=NARR][AGR=3SG]]
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]?
```

## A.2 Non-Finite Clauses

```
;;;============
;;; infinitive
;;;============
;; cam kIrmak
(uni '((cat simple-clause)
       (mood infinitive) (non-finite mek)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants ((medium  ((cat common)
                                (definite no)
                                (lex "cam")))))))
;;[[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=NOM]]
;;[[CAT=VERB][ROOT=kIr][SENSE=POS][CONV=NOUN=MEK]
;;  [TYPE=INFINITIVE][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; camIn ali tarafindan kIrIlmasI
(uni '((cat simple-clause)
       (mood infinitive) (non-finite me)
       (voice passive)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex "kIr")))
       (participants ((actor ((cat proper)
                              (lex "ali")))
                      (medium  ((cat common)
                                (definite yes)
                                (lex "cam")))))))
;;  [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=GEN]]
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;; tarafIndan
;;  [[CAT=VERB][ROOT=kIr][VOICE=PASS][SENSE=POS][CONV=NOUN=ME]
;;   [TYPE=INFINITIVE][AGR=3SG][POSS=3SG][CASE=NOM]].
;;=============
;; ben camIn ali tarafIndan kIrIlmasInI  istedim.
```

```
(uni '((cat simple-clause)
       (time past)
       (mood declarative) (voice active)
       (process ((type mental) (type-of-base verb)
                 (lex "iste")))
       (participants ((senser
                       ((cat pronoun)
                        (person first) (number singular)))
                      (phenomenon
                       ((cat simple-clause)
                        (mood infinitive) (non-finite me)
                        (voice passive)
                        (process ((type material)
                                  (type-of-base verb)
                                  (agentive yes)
                                  (effective yes)
                                  (lex "kIr")))
                        (participants
                          ((actor  ((cat proper)
                                     (lex "ali")))
                           (medium  ((cat common)
                                      (definite yes)
                                      (lex "cam")))))))))))
;;  [[CAT=NOUN][ROOT=ben][AGR=1SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=GEN]]
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;; tarafIndan
;;  [[CAT=VERB][ROOT=kIr][VOICE=PASS][SENSE=POS][CONV=NOUN=ME]
;;   [TYPE=INFINITIVE][AGR=3SG][POSS=3SG][CASE=ACC]]
;;  [[CAT=VERB][ROOT=iste][SENSE=POS][TAM1=PAST][AGR=1SG]].

;;;============
;;; participle
;;;============
;; turgay'In camI kIrdIGI  .....[okul]....
(uni '((cat simple-clause)
       (mood participle) (participle past)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex-transition transitive)
                 (lex "kIr")))
      (scope {^ circum location})
       (participants ((agent ((cat proper)
                              (lex "Turgay")))
                      (medium  ((cat common)
                                (definite yes)
                                (lex "cam")))))
       (circum ((location ((lex "okul")))))))
;;  [[CAT=NOUN][ROOT=Turgay][AGR=3SG][POSS=NONE][CASE=GEN]]
;;  [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACC]]
;;  [[CAT=VERB][ROOT=kIr][SENSE=POS][CONV=ADJ=DIGI][POSS=3SG]].
;;=-=-=-=-=-=
;; turgay'In camI kIrdIGI okul
(uni '((cat common)
       (lex "ev")
       (qualifier
        ((qual-desc ((cat simple-clause)
                     (mood participle) (participle past)
                     (process ((type material)
                               (type-of-base verb)
                               (agentive yes) (effective yes)
                               (lex-transition transitive)
                               (lex "kIr")))
                     (scope {^ circum location })
                     (participants
                       ((actor ((cat proper)
                                (person third) (number plural)
                                (lex "turgay")))
                        (medium  ((cat common)
                                  (definite yes)
                                  (lex "cam")))))
                     (circum ((location ((lex "okul"))))))))))
;;  [[CAT=NOUN][ROOT=turgay][AGR=3PL][POSS=NONE][CASE=GEN]]
;;  [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACC]]
;;  [[CAT=VERB][ROOT=kIr][SENSE=POS][CONV=ADJ=DIGI][POSS=3SG]]
;;  [[CAT=NOUN][ROOT=ev][AGR=3SG][POSS=NONE][CASE=NOM]].

;;;============
;;; adverbial
;;;============
;; ali'nin camI kIrarak
(uni '((cat simple-clause)
       (mood adverbial) (non-finite arak)
       (process ((type material) (type-of-base verb)
                 (agentive yes)  (effective yes)
                 (lex-transition transitive)
                 (lex "kIr")))
       (participants ((actor ((cat proper)
                              (lex "ali")))
                      (medium  ((cat common)
                                (definite yes)
                                (lex "cam")))))))
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACC]]
;;  [[CAT=VERB][ROOT=kIr][SENSE=POS][CONV=ADVERB=ARAK]].
;;=-=-=-=-=-=
;; ali'nin camI kIrarak girdigi ev
(uni '((cat common)
       (lex "ev")
       (qualifier
        ((qual-desc
          ((cat simple-clause)
           (mood participle)  (participle past)
           (process ((type material) (type-of-base verb)
                     (agentive yes)  (effective no)
                     (lex "gir")))
           (scope {^ circum location })
           (participants
            ((actor ((cat proper)
                     (lex "ali")))))
           (circum
            ((location ((lex "ev")))
             (quality
              ((cat simple-clause)
               (mood adverbial) (non-finite arak)
               (process ((type material) (type-of-base verb)
                         (agentive yes)  (effective yes)
                         (lex "kIr")))
               (participants
                ((medium  ((cat common)
                           (definite yes)
                           (lex "cam"))))))))) ))))))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACC]]
;; [[CAT=VERB][ROOT=kIr][SENSE=POS][CONV=ADVERB=ARAK]]
;; [[CAT=VERB][ROOT=gir][SENSE=POS][CONV=ADJ=DIGI][POSS=3SG]]
;; [[CAT=NOUN][ROOT=ev][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=-=-=-=-=-=
;; Ali'nin camI kIrarak girdigi evde UC Cocuk varmIS.
(uni '((cat simple-clause)
       (time narr)
       (mood declarative) (voice active)
       (process ((type existential)
                 (existential yes)
                 (polarity positive)))
       (topic {^ circum location})
       (participants
        ((entity ((cat common)
                  (lex "Cocuk")
                  (quantitative
                    ((exact yes)
                     (value ((lex "UC")))
                     (quan-type cardinal)))))))
       (circum
        ((location
          ((cat common)
           (lex "ev")
           (qualifier
          ((qual-desc
            ((cat simple-clause)
             (mood participle) (participle past)
             (process ((type material)
                       (type-of-base verb)
                       (agentive yes)  (effective no)
                       (lex-transition transitive)
                       (lex "gir")))
             (scope {^ circum location })
             (participants
              ((actor ((cat proper)
                       (lex "ali")))))
             (circum
              ((location ((lex "ev")))
               (quality
                ((cat simple-clause)
                 (mood adverbial) (non-finite arak)
                 (process ((type material) (type-of-base verb)
                           (agentive yes) (effective yes)
                           (lex-transition transitive)
                           (lex "kIr")))
                 (participants
                  ((medium  ((cat common)
                             (definite yes)
                             (lex "cam"))))))))))))))))))
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;;  [[CAT=NOUN][ROOT=cam][AGR=3SG][POSS=NONE][CASE=ACC]]
;;  [[CAT=VERB][ROOT=kIr][SENSE=POS][CONV=ADVERB=ARAK]]
;;  [[CAT=VERB][ROOT=gir][SENSE=POS][CONV=ADJ=DIGI][POSS=3SG]]
;;  [[CAT=NOUN][ROOT=ev][AGR=3SG][POSS=NONE][CASE=LOC]]
;;  [[CAT=ADJ][ROOT=UC]]
;;  [[CAT=NOUN][ROOT=Cocuk][AGR=3SG][POSS=NONE][CASE=NOM]]
;;  [[CAT=NOUN][ROOT=var][AGR=3SG][POSS=NONE][CASE=NOM]
;;   [CONV=VERB=NONE][TAM1=NARR][AGR=3SG]].
```

## A.3 Noun Group (NP)

```
;;=================
;; Noun Group
;;=================
;; ali'nin kitabI
(uni   '((cat common)
         (lex "kitap")
         (possessor ((number singular)
                     (person third)
                     (cat proper) (lex "ali")))))
;;  [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;;  [[CAT=NOUN][ROOT=kitap][AGR=3SG][POSS=3SG][CASE=NOM]].
;;=-=-=-=-=-=
;; kimin kitabI
(uni   '((cat common)
         (lex "kitap")
         (possessor ((type interrogative)))))
;;  [[CAT=NOUN][ROOT=kim][AGR=3SG][POSS=NONE][CASE=GEN]]
;;  [[CAT=NOUN][ROOT=kitap][AGR=3SG][POSS=3SG][CASE=NOM]].
;;=-=-=-=-=-=
;; hangi kitap
(uni   '((cat common)
         (lex "kitab")
         (demonstrative ((type interrogative)))))
;;  [[CAT=ADJ][ROOT=hangi]]
;;  [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; Su kitap
(uni   '((cat common)
         (lex "kitap")
         (demonstrative ((distance far)))))
;;  [[CAT=ADJ][ROOT=Su]]
;;  [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; ali'nin hangi kitabI
```

```
(uni   '((cat common)
         (lex "kitap")
         (possessor ((number singular)
                     (person third)
                     (cat proper) (lex "ali")))
         (demonstrative ((type interrogative)))))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=ADJ][ROOT=hangi]]
;; [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=3SG][CASE=NOM]].
;;=-=-=-=-=-=
;; ali'nin Su ilk beS kitabI
(uni   '((cat common)
         (lex "kitab")
         (possessor ((number singular)
                     (person third)
                     (cat proper) (lex "ali")))
         (demonstrative ((type determinative)
                         (distance far)))
         (quantitative  ((exact yes)
                         (quan-type cardinal)
                         (value ((lex "beS")))))
         (ordinal ((lex "ilk")))))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=ADJ][ROOT=Su]]
;; [[CAT=ADJ][ROOT=ilk]]
;; [[CAT=ADJ][ROOT=beS]]
;; [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=3SG][CASE=NOM]].
;;=-=-=-=-=-=
;; Su ilginC beS eski bUyUk kitap
(uni   '((cat common)
         (lex "kitap")
         (specific yes)
         (demonstrative ((type determinative)
                         (distance far)))
         (quantitative  ((exact yes)
                         (quan-type cardinal)
                         (value ((lex "beS")))))
```

```
          (epithet ((attitude ((lex "ilginC")))
                    (quality ((age
                               ((lex "eski")))
                              (size
                               ((lex "bUyUk")))))))))
;; [[CAT=ADJ][ROOT=Su]]
;; [[CAT=ADJ][ROOT=ilginC]]
;; [[CAT=ADJ][ROOT=beS]]
;; [[CAT=ADJ][ROOT=eski]]
;; [[CAT=ADJ][ROOT=bUyUk]]
;; [[CAT=NOUN][ROOT=kitap][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; bUtun kitaplar
(uni '((cat common)
       (lex "kitap")
       (specific no)
       (countability countable)
       (number plural)
       (ns-deictic ((total +)))))
;; [[CAT=ADJ][ROOT=bUtUn]]
;; [[CAT=NOUN][ROOT=kitap][AGR=3PL][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; baSka bir kitap
;; baSka herhangi bir kitap
;; diGer bir kitap ...
(uni '((cat common)
       (lex "kitap")
       (specific no)
       (ns-deictic ((additional yes)
                    ;;(total -)
                    (partial yes)))))
;; [[CAT=ADJ][ROOT=diGer]]
;; [[CAT=ADJ][ROOT=herhangi bir]]
;; [[CAT=NOUN][ROOT=kitap][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; ali'nin bazI kitablarI
(uni '((cat common)
       (lex "kitap")
       (definite no)
       (specific yes)
       (possessor ((number singular) (person third)
                   (cat proper)      (lex "ali")))
       (number plural)
       (ns-deictic ((partial yes)))))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=ADJ][ROOT=bazI]]
;; [[CAT=NOUN][ROOT=kitap][AGR=3PL][POSS=3SG][CASE=NOM]].
;;=-=-=-=-=-=
;; yedi tane elma
;; yedi adet elma
(uni '((cat common)
       (lex "elma")
       (definite no)
       (specific no)
       (countability countable)
       (number singular)
       (quantitative ((exact yes)
                      (value ((lex "yedi")))
                      (quan-type cardinal)
                      (partitive yes)
                      (part-type  measure)
                      (measure number)))))
;; [[CAT=ADJ][ROOT=yedi]]
;; [[CAT=NOUN][ROOT=adet][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=elma][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; 2 kasa dolusu elma
(uni '((cat common)
       (lex "elma")
       (definite no)
       (specific no)
       (countability countable)
       (number singular)       ; plural)
       (quantitative ((exact yes)
                       (value ((lex "iki")))
                       (quan-type cardinal)
                       (partitive yes)
                       (part-type  measure)
                       (measure container)
                       (container ((lex "kasa")))
                       (full yes)))))
;; [[CAT=ADJ][ROOT=iki]]
;; [[CAT=NOUN][ROOT=kasa][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=dolusu][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=elma][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; iki dilim kek
;; iki parca kek
(uni '((cat common)
       (lex "kek")
       (definite no)
       (specific no)
       (countability mass)
       (quantitative ((exact yes)
                      (value ((lex "iki")))
                      (quan-type cardinal)
                      (partitive yes)
                      (part-type typical)))))
;; [[CAT=ADJ][ROOT=iki]]
;; [[CAT=NOUN][ROOT=dilim][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=kek][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; en Cok 10 dilim/parca kek
;; en fazla 10  dilim/parca kek
;; en az 10 dilim/parca kek
;; yakalSIk/ortalama 10 dilim/parca kek
(uni '((cat common)
       (lex "kek")
       (definite no)
       (specific no)
       (countability mass)
       (quantitative ((exact no)
                       (quan-type max)
                       (max ((lex "10")))
                       ;; (quan-type min)
                       ;;  (min ((lex "10")))
                       ;; (quan-type approx)
                       ;;  (approx ((lex "10")))
                       (partitive yes)
                       (part-type typical)))))
;; [[CAT=ADVERB][ROOT=en]]
;; [[CAT=ADJ][ROOT=Cok]]
;; [[CAT=ADJ][ROOT=10]]
;; [[CAT=NOUN][ROOT=dilim][AGR=3SG][POSS=NONE][CASE=NOM]]
;; [[CAT=NOUN][ROOT=kek][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; ali'nin Su kitablarInIn hepsi/tamamI
(uni   '((cat common)
         (lex "kitab")
         (number plural)
         (possessor ((number singular)
                     (person third)
                     (cat proper) (lex "ali")))
         (demonstrative ((type determinative)
                         (distance far)))
         (post-det ((type total)
                    (total +)))))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=ADJ][ROOT=Su]]
;; [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=hep][AGR=3SG][POSS=3SG][CASE=NOM]].
;;=-=-=-=-=-=
;; kitablarIn beS katI
(uni   '((cat common)
         (lex "kitab")
         (specific no)
         (definite no)
         (number plural)
         (post-det ((type multiplier)
                    (multiplier ((lex "beS")))))))
;; [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=ADJ][ROOT=beS]]
;; [[CAT=NOUN][ROOT=kat][AGR=3SG][POSS=3SG][CASE=NOM]].
;;=-=-=-=-=-=
;; elmanIn beSte dOrdU
(uni    '((cat common)
         (lex "elma")
         (specific no)
         (definite no)
         (number singular)
         (post-det ((type fraction)
                    (numerator ((lex "beS")))
                    (denumerator ((lex "dOrt")))))))
;; [[CAT=NOUN][ROOT=elma][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=beS][AGR=3SG][POSS=NONE][CASE=LOC]]
;; [[CAT=NOUN][ROOT=dOrt][AGR=3SG][POSS=3SG][CASE=NOM]].
;;------------------------------
;; w i t h   q u a l i f i e r
;;------------------------------
;; kIrmIzI SapkalI iki kIz
(uni    '((cat common)
         (lex "kIz")
         (specific no)
         (definite no)
         (number singular)
         (quantitative ((exact yes)
                        (value ((lex "iki")))
                        (quan-type cardinal)))
         (qualifier
          ((possession ((it-is +)
                        (lex "Sapka")
                        (cat common)
                        (specific no)
                        (definite no)
                        (number singular)
```

```
                         (epithet
                          ((quality
                            ((color ((type exact)
                                     (lex "kIrmIzI"))))))))))))
;; [[CAT=ADJ][ROOT=kIrmIzI]]
;; [[CAT=NOUN][ROOT=Sapka][AGR=3SG][POSS=NONE][CASE=MUNITIVE]]
;; [[CAT=ADJ][ROOT=iki]]
;; [[CAT=NOUN][ROOT=kIz][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; ali'nin elindeki kitap
(uni   '((cat np)
         (cat common)
         (lex "kitap")
         (specific no)
         (definite no)
         (number singular)
         (qualifier
          ((qual-loc ((it-is spatial)
                      (lex "el")
                      (cat common)
                      (specific yes)
                      (definite no)
                      (number singular)
                      (possessor ((number singular)
                                  (person third)
                                  (cat proper) (lex "ali")))))))))
;; [[CAT=NOUN][ROOT=ali][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=el][AGR=3SG][POSS=3SG][CASE=LOC][CONV=ADJ=REL]]
;; [[CAT=NOUN][ROOT=kitap][AGR=3SG][POSS=NONE][CASE=NOM]].
;;=-=-=-=-=-=
;; ahmet'in kardeSine ait  kitaplar
(uni   '((cat np)
         (cat common)
         (lex "kitap")
         (specific no)
         (definite no)
         (number plural)
         (qualifier
          ((qual-possessor
            ((np ((cat common)
                  (lex "kardeS")
                  (specific yes)
                  (definite no)
                  (possessor ((number singular)
                              (person third)
                              (cat proper)
                              (lex "ahmet"))))))))))

;; [[CAT=NOUN][ROOT=ahmet][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=kardeS][AGR=3SG][POSS=3SG][CASE=DAT]]
;; ait
;; [[CAT=NOUN][ROOT=kitap][AGR=3PL][POSS=NONE][CASE=NOM]].

;;============================
;; Postpositional Group
;;============================
;; ahmet'in kardesinden sonra
(uni '((cat pp)
       (post-pos ((lex "sonra")
                  (subcat dative)))
       (np ((cat common)
            (lex "kardeS")
            (specific yes)
            (definite no)
            (number singular)
            (possessor ((number singular)
                        (person third)
                        (cat proper)
                        (lex "ahmet")))))))
;; [[CAT=NOUN][ROOT=ahmet][AGR=3SG][POSS=NONE][CASE=GEN]]
;; [[CAT=NOUN][ROOT=kardeS][AGR=3SG][POSS=3SG][CASE=DAT]]
;; sonra.
```

# Appendix B

# Non-finite Elements in the Participles

This section presents the linguistic analysis of the participles according to the semantic function of the displaced constituent, and some features such as *time*, *voice*, *transition* of the process. The following tables describe the several exceptions in the determination of the non-finite element. In the tables, *time* represents the selected time for the participle; *Voice* represents the selected voice for the process; *Trans* represents the transitive feature of the process; **non-f** represents the relevant non-finite element to realize the participle; *Poss* represents the possessor of the semantic function that can be also displaced.

| Time | Voice | Trans. | non-f. | spec/non-spec D. Obj | Poss. |
|---|---|---|---|---|---|
| Past | Active | intrans | -miş | uyumuş **çocuk**$_{actor/medium}$ | + |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | camı/cam kırmış **çocuk**$_{agent/actor}$ | + |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | caus. | -miş | camı/cam kırdırmış **çocuk**$_{agent}$ | + |
| | | | -dik | —/cam kırdırmadık **çocuk**$_{actor}$ | + |
| | | | -diği | camı/cam kırdırdığım **çocuk**$_{actor}$ | + |
| | Passive | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | caus. | -miş | —/cam kırdırılmış **çocuk**$_{actor}$ | + |
| | | | -dik | —/cam kırdırılmadık **çocuk**$_{actor}$ | + |
| | | | -diği | — | — |
| Present | Active | intrans | -en | uyuyan **çocuk**$_{actor/medium}$ | + |
| | | trans | -en | camı/cam kıran **çocuk**$_{agent/actor}$ | + |
| | | caus. | -en | camı/cam kırdıran **çocuk**$_{agent}$ | + |
| | Passive | intrans | -en | — | — |
| | | trans | -en | — | — |
| | | caus | -en | —/cam kırdırılan **çocuk**$_{actor}$ | + |
| Future | Active | intrans | -ecek | uyuyacak **çocuk**$_{actor/medium}$ | + |
| | | | -eceği | — | — |
| | | trans | -ecek | camı/cam kıracak **çocuk**$_{agent/actor}$ | + |
| | | | -eceği | — | — |
| | | caus. | -ecek | camı/cam kırdıracak **çocuk**$_{agent}$ | + |
| | | | -eceği | camı/cam kırdıracağım **çocuk**$_{actor}$ | + |
| | Passive | intrans | -ecek | — | — |
| | | | -eceği | — | — |
| | | trans | -ecek | — | — |
| | | | -eceği | — | — |
| | | caus. | -ecek | —/cam kırdırılacak **çocuk**$_{actor}$ | + |
| | | | -eceği | camın/— kırdırılacağı **çocuk**$_{actor}$ | — |

Table B.1: Participle forms that modify AGENT or ACTOR

| Time | Voice | Trans. | non-f. | conflated Actor or Goal | Poss. |
|---|---|---|---|---|---|
| Past | Active | intrans | -miş | suya batmış **gemi**$_{actor}$ / — | + |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | — | — |
| | | | -dik | — / kırmadık **odun**$_{goal}$ | + |
| | | | -diği | — / kırdığım **odun**$_{goal}$ | + |
| | | caus. | -miş | — / — | — |
| | | | -dik | — / kırdırmadık **odun**$_{goal}$ | + |
| | | | -diği | — / kırdırdığım **odun**$_{goal}$ | + |
| | Passive | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | — / kırılmış **odun**$_{goal}$ | + |
| | | | -dik | — / kırılmadık **odun**$_{goal}$ | + |
| | | | -diği | — | — |
| | | caus. | -miş | — / kırdırılmış **odun**$_{goal}$ | + |
| | | | -dik | — / kırdırılmadık **odun**$_{goal}$ | — |
| | | | -diği | — | — |
| Present | Active | intrans | -en | suya batan **gemi**$_{actor}$ / — | + |
| | | trans | -en | — | — |
| | | caus. | -en | — | — |
| | Passive | intrans | -en | — | — |
| | | trans | -en | — / kırılan **odun**$_{goal}$ | + |
| | | caus | -en | — / kırdırılan **odun**$_{goal}$ | + |
| Future | Active | intrans | -ecek | suya batacak **gemi**$_{actor}$ / — | + |
| | | | -eceği | — | — |
| | | trans | -ecek | — / kıracak **odun**$_{goal}$ | + |
| | | | -eceği | — / kıracağım **odun**$_{goal}$ | + |
| | | caus. | -ecek | — / kırdıracak **odun**$_{goal}$ | + |
| | | | -eceği | — / kırdıracağım **odun**$_{goal}$ | + |
| | Passive | intrans | -ecek | — | — |
| | | | -eceği | — | — |
| | | trans | -ecek | — / kırılacak **odun**$_{goal}$ | + |
| | | | -eceği | — | — |
| | | caus. | -ecek | — / kırdırılacak **odun**$_{goal}$ | + |
| | | | -eceği | — | — |

Table B.2: Participle forms that modify MEDIUM (conflated with Actor or Goal)

| Time | Voice | Trans. | non-f. | spec/non-spec D. Obj | Poss. |
|---|---|---|---|---|---|
| Past | Active | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | — | — |
| | | | -dik | —/para vermedik **çocuk** | + |
| | | | -diği | odunu/odun kırdığım **kadın** | + |
| | | caus. | -miş | —/— | — |
| | | | -dik | —/para verdirmedik **çocuk** $?_{actor}$ | + |
| | | | -diği | odunu/odun kırdırdığım **kadın** $?_{actor}$ | + |
| | Passive | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | —/para verilmiş **çocuk** | + |
| | | | -dik | —/para verilmedik **çocuk** | + |
| | | | -diği | paranın/— verildiği **çocuk** | — |
| | | caus. | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| Present | Active | intrans | -en | — | — |
| | | trans | -en | — | — |
| | | caus. | -en | — | — |
| | Passive | intrans | -en | — | — |
| | | trans | -en | —/odun kırılan **adam** | + |
| | | caus | -en | — | — |
| Future | Active | intrans | -ecek | — | — |
| | | | -eceği | — | — |
| | | trans | -ecek | — | — |
| | | | -eceği | parayı/para vereceğim **çocuk** | + |
| | | caus. | -ecek | — | — |
| | | | -eceği | parayı/para verdireceğim **çocuk** $?_{actor}$ | + |
| | Passive | intrans | -ecek | — | — |
| | | | -eceği | — | — |
| | | trans | -ecek | —/odun kırılacak **kadın** | + |
| | | | -eceği | — | — |
| | | caus. | -ecek | —/odun kırdırılacak **kadın** $?_{actor}$ | + |
| | | | -eceği | — | — |

Table B.3: Participle forms that modify BENEFICIARY

| Time | Voice | Trans. | non-f. | spec/non-spec D. Obj | Poss. |
|---|---|---|---|---|---|
| Past | Active | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | geminin/gemi battığı **gün/süre** | +/- |
| | | trans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | odunu/odun kırdığım **gün/süre** | +/- |
| | | caus. | -miş | —/— | — |
| | | | -dik | — | — |
| | | | -diği | odunu/odun kırdırdığım **gün/süre** | +/- |
| | Passive | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | paranın/para harcandığı **gün/süre** | +/- |
| | | caus. | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| Present | Active | intrans | -en | — ? uçak düşen gün/— | — |
| | | trans | -en | — | — |
| | | caus. | -en | — | — |
| | Passive | intrans | -en | — | — |
| | | trans | -en | —/para harcanan **gün**/— | +/- |
| | | caus | -en | —/para harcatılan **gün**/— | +/- |
| Future | Active | intrans | -ecek | — | — |
| | | | -eceği | geminin/gemi batacağı **gün/süre** | +/- |
| | | trans | -ecek | —/odun kıracak **gün/süre** | +/- |
| | | | -eceği | parayı/para harcayacağım **gün/süre** | +/- |
| | | caus. | -ecek | — | — |
| | | | -eceği | odunu/odun kırdıracağım **gün/süre** | + |
| | Passive | intrans | -ecek | — | — |
| | | | -eceği | — | — |
| | | trans | -ecek | —/odun kırılacak **gün/süre** | +/- |
| | | | -eceği | odunun/odun kırılacağı **gün/süre** | — |
| | | caus. | -ecek | —/odun kırdırılacak **gün/süre** | +/- |
| | | | -eceği | odunun/odun kırılacağı **gün/süre** | — |

Table B.4: Participle forms that modify TIME or DURATION

| Time | Voice | Trans. | non-f. | spec/non-spec D. Obj | Poss. |
|---|---|---|---|---|---|
| Past | Active | intrans | -miş | çocuk/— uyumuş **yatak** | + |
| | | | -dik | — | — |
| | | | -diği | çocuğun/— uyuduğu **ev** | + |
| | | trans | -miş | — | — |
| | | | -dik | —/top oynamadık **bahçe** | + |
| | | | -diği | odunu/odun kesdiğim **orman** | + |
| | | caus. | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | çocuğu/— ata bindirdiğim **tarla** | + |
| | Passive | intrans | -miş | (çocuk tarafından) uyunmuş **yatak** | + |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | paranın/— verildiği **yer** | + |
| | | caus. | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | odunu/odun kestirdiğim **bahçe** | + |
| Present | Active | intrans | -en | —/çocuk uyuyan **yatak** | + |
| | | trans | -en | — | — |
| | | caus. | -en | — | — |
| | Passive | intrans | -en | —/— uyunan **yatak** | + |
| | | trans | -en | —/odun kırılan **bahçe** | + |
| | | caus | -en | —/odun kırdırılan **yer** | + |
| Future | Active | intrans | -ecek | —/çocuk yatacak **yatak** | + |
| | | | -eceği | çocuğun yatacağı **yatak** | + |
| | | trans | -ecek | —/top oynayacak **saha** | — |
| | | | -eceği | parayı/para harcayacağım **köy** | + |
| | | caus. | -ecek | — | — |
| | | | -eceği | çocuğa dersi/ders anlattıracağım **oda** | + |
| | Passive | intrans | -ecek | uyuyacak **yatak** | + |
| | | | -eceği | çocuğun uyuyacağı **yatak** | + |
| | | trans | -ecek | —/odun kırılacak **bahçe** | + |
| | | | -eceği | odunun/— kırılacağı **bahçe** | + |
| | | caus. | -ecek | —/odun kırdırılacak **bahçe** | + |
| | | | -eceği | odunun/— kırdırılacağı **bahçe** | + |

Table B.5: Participle forms that modify LOCATION

| Time | Voice | Trans. | non-f. | spec/non-spec D. Obj | Poss. |
|---|---|---|---|---|---|
| Past | Active | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | çocuğun/— gittiği/geldiği **ev** | + |
| | | trans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | adamın kumu/kum döktüğü **deniz** | + |
| | | caus. | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | adamın kumu/kum döktürdüğü **deniz** | + |
| | Passive | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | kumun/— döküldüğü **deniz** | + |
| | | caus. | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | kumun/— döktürtüldüğü **deniz** | + |
| Present | Active | intrans | -en | — | + |
| | | trans | -en | — | — |
| | | caus. | -en | — | — |
| | Passive | intrans | -en | gidilen/gelinen **ev** $?_{Origin/Destination}$ | + |
| | | trans | -en | —/kum dökülen/çıkarılan **deniz** | + |
| | | caus | -en | —/kum döktürülen **deniz** | + |
| Future | Active | intrans | -ecek | — | — |
| | | | -eceği | gideceğim/geleceğim **ev** | + |
| | | trans | -ecek | —/kum dökecek/çıkaracak **deniz** | + |
| | | | -eceği | kumu/kum dökeceğim **deniz** | + |
| | | caus. | -ecek | —/kum döktürecek **deniz** | + |
| | | | -eceği | —/kum döktüreceğim **deniz** | + |
| | Passive | intrans | -ecek | gidilecek/gelinecek **ev** | + |
| | | | -eceği | — | + |
| | | trans | -ecek | —/kum dökülecek **deniz** | + |
| | | | -eceği | kumun/— döküleceği **deniz** | + |
| | | caus. | -ecek | —/kum döktürülecek **deniz** | + |
| | | | -eceği | kumun/— döktürüleceği **deniz** | + |

Table B.6: Participle forms that modify ORIGIN and DESTINATION

| Time | Voice | Trans. | non-f. | spec/non-spec D. Obj | Poss. |
|---|---|---|---|---|---|
| Past | Active | intrans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | çocuğun/— koştuğu **son üç km** | + |
| | | trans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | adamın bayrağı/bayrak taşıdığı **3 km** | + |
| | | caus. | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | adamın bayrağı/bayrak taşıttığı **3 km** | + |
| | Passive | intrans | -miş | (x tarafından) koşulmuş **son üç km** | + |
| | | | -dik | — | — |
| | | | -diği | — | — |
| | | trans | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | bayrağın/— taşındığı **son yüz metre** | — |
| | | caus. | -miş | — | — |
| | | | -dik | — | — |
| | | | -diği | koşturulduğum **beş km** | — |
| Present | Active | intrans | -en | — | + |
| | | trans | -en | — | — |
| | | caus. | -en | — | — |
| | Passive | intrans | -en | koşulan **beş km** | + |
| | | trans | -en | —/bayrak taşınan **üç km** | — |
| | | caus | -en | —/bayrak taşıtılan **üc km** | — |
| Future | Active | intrans | -ecek | koşacak **5 km** | — |
| | | | -eceği | çocuğun koşacağı **5 km** | — |
| | | trans | -ecek | —/bayrak taşıyacak **son 3 km** | — |
| | | | -eceği | bayrağı/bayrak taşıyacağım **3 km** | — |
| | | caus. | -ecek | taşıttıracak **son 3 km** | — |
| | | | -eceği | taşıttıracağım **son 3 km** | — |
| | Passive | intrans | -ecek | koşulacak **5 km** | — |
| | | | -eceği | — | — |
| | | trans | -ecek | —/bayrak taşınacak **son 3 km** | — |
| | | | -eceği | bayrağın/— taşınacağı **son 3 km** | — |
| | | caus. | -ecek | —/bayrak taşıttırılacak **son 3 km** | — |
| | | | -eceği | bayrağın/— taşıttırılacağı **son 3 km** | — |

Table B.7: Participle forms that modify DISTANCE

# Bibliography

[1] D.E. Appelt. *Planning Natural Language Utterances.* Cambridge University Press, Cambridge, 1985.

[2] T. Banguoğlu. *Türkçenin Grameri.* Number 528 in Türk Dil Kurumu Yayınları. Türk Tarih Kurumu Basım Evi, Ankara, 1986.

[3] J.A. Bateman, M. Emele, and S. Momma. The nondirectional representation of systemic functional grammars and semantics as typed feature structures. In *Proceeding of COLING-92*, pages 916–920, August 23-28 1992.

[4] R. Dale. A short annotated bibliography of research in natural language generation. August 1990.

[5] A. Davey. *Discourse Production: A Computer Model of Some Aspects of a Speaker.* Edinburgh University Press, Edinburgh, 1978.

[6] M. Elhadad. Types in functional unification grammars. In *Proceedings of ACL'90*, pages 642–655, 1990.

[7] M. Elhadad. *Using Argumentation to Control Lexical Choice: a Functional Unification Based Approach.* PhD thesis, Department of Computer Science, Columbia University, 1990.

[8] M. Elhadad. *FUF: the Universal Unifier User manual 5.2.* Department of Computer Science, Ben Gurion University of the Negev, June 1993.

[9] M. Elhadad and R. Jacques. An overview of SURGE: A reusable comprehensive syntactic realization component. 1990.

[10] E.E. Erguvanlı. *The Function of Word Order in Turkish Grammar.* PhD thesis, University of California, Los Angeles, 1979.

[11] T. Givon. *English Grammar - A Function-Based Introduction*, volume I and II. John Benjamins Publishing Company, Linguistic Dept. University of Oregon, 1993.

[12] N. Goldman. *Computer Generation of Natural Language from a Deep Conceptual Base.* PhD thesis, Yale University, 1974.

[13] M. A. K. Halliday. Language structure and language function. In John Lyons, editor, *New Horizons in Linguistics*, chapter 7, pages 140–165. Penguin Books, 1970.

[14] M. A. K. Halliday. *An Introduction to Functional Grammar.* Edward Arnold, London, 1985.

[15] B. Hoffman. *The Computational Analysis of the Syntax and Interpretation of "Free" Word Order in Turkish.* PhD thesis, University of Pennsylvania, June 1995.

[16] E.H. Hovy. Pragmatics and natural language generation. *Artificial Intelligence*, pages 43:153–197, 1990.

[17] R. Jacques. *Revision-based generation of natural language summaries providing historical background: corpus-based analysis, design, implementation and evaluation.* PhD thesis, Department of Computer Science, Columbia University, New York, USA, 1994.

[18] M. Kantrowitz and J. Bates. Integrated natural language generation systems. Technical Report CMU-CS-92-107, School of Computer Science, Carnegie Mellon University, Pittsburgh, April 1992.

[19] L. Karahan. *Türkçe Söz Dizimi - Cümle Tahlilleri -.* Kaynak Yayınlar. Akçağ Yayınları, Ankara, 1993.

[20] R.T. Kasper. *Systemic Grammar and Functional Unification Grammar*, chapter 9, pages 176–199. In Systemic Functional Approaches to Discourse. Ablex, 1988.

[21] R.T. Kasper and William C. Rounds. A logical semantics for feature structures. In *Proceedings of the 24th meeting of the ACL*, pages 257–266, June 1986.

[22] M. Kay. Functional grammar. In *Proceedings of the 5th Meeting of the Berkeley Linguistics Society*, pages 142–58. Berkeley Linguistics Society, 1979.

[23] N. Koç. *Yeni Dilbilgisi.* İnkilap Kitapevi, İstanbul, 1990.

[24] Z. Korkmaz. *Gramer Terimleri Sözlüğü.* Number 575 in Türk Dil Kurumu Yayınları. Türk Tarih Kurumu Basım Evi, Ankara, 1992.

[25] McKeown K.R. and M. Elhadad. A contrastive evaluation of functional unification grammar for surface language generation: A case study in choice of connectives. In C.L. Paris, W.R. Swartout, and W.C. Mann, editors. *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, chapter 14, pages 351–396. Kluwer Academic, 1991.

[26] T. Kumano, T. Tokunga, K. Inui, and H. Tanaka. GENESYS: An integrated environment for developing systemic functional grammars. In *Proceddings of the International Workshop on Sharable Natural Language Resources*, pages 78–85, Nara Institute of Science and Technology Ikoma, Nara, Japan, 10 - 11 August 1994.

[27] C. M. Matthiessen. Notes on the organization of the environment of a text generation grammar. In Gerard Kempen, editor, *Natural Language Generation*, number 135 in Applied Sciences, chapter 17, pages 253–278. Martinus Nijhoff, 1987.

[28] C. M. Matthiessen. Lexico(grammatical) choice in text generation. In C.L. Paris, W.R. Swartout, and W.C. Mann, editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, chapter 10, pages 249–292. Kluwer Academic, 1991.

[29] C. M. Matthiessen and J. A. Bateman. Nigel: A systemic grammar for text generation. Technical Report ISI/RR-83-105, Information Sciences Institute, University of Southern California, February 1983.

[30] C. M. Matthiessen and J. A. Bateman. *Text Generation and Systemic-Functional Linguistic.* Communication in Artificial Intelligence Series. Pinter Publishers, 1991.

[31] D. D. McDonald. *Natural Language Production as a Process of Decision Making under Constraint.* PhD thesis, MIT, Cambridge, MA, 1980.

[32] D. D. McDonald. Natural language generation. *Encyclopedia of Artificial Intelligence*, pages 642–655, 1987.

[33] K.R. McKeown. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.* Cambridge University Press, Cambridge, 1985.

[34] M. O'Donnell. *Sentence Analysis and Generation – a Systemic Perspective.* PhD thesis, Linguistics Dept., University of Sydney, 1994.

[35] M. O'Donnell. Sentence generation using the systemic workbench. In *Proceedings of the Fifth European Workshop on Natural Language Generation*, pages 235–238., Leiden, The Netherlands, 20-22 May 1995.

[36] K. Oflazer. Two-level description of Turkish morphology. In *In Proceedings of the Sixth Conferance of the European Chapter of the Association for Computational Linguistic.* April 1993.

[37] T. Patten. *Systemic Text Generation as Problem Solving.* Studies in Natural Language Processing. Cambridge University Press, 1988.

[38] R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A Grammar of Contemporary English.* Longman, London, twentieth edition, 1992.

[39] H. I. Sebuktekin. *Turkish-English contrastive analysis : Turkish morphology and corresponding English structures.* Janua linguarum. Series practica 84. 1971.

[40] T. Winograd. *Understanding Natural Language.* Acedemic Press, New York and London, 1972.

[41] T. Winograd. *Language as a Cognitive Process.* Addison-Wesley, London, 1983.